Titre: Title:	A Model for Software Quality Evaluation Using the User's Point of Views
Auteur: Author:	Seyed Reza Mirsalari
Date:	2017
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Mirsalari, S. R. (2017). A Model for Software Quality Evaluation Using the User's Point of Views [Ph.D. thesis, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/2813/

UNIVERSITÉ DE MONTRÉAL

A MODEL FOR SOFTWARE QUALITY EVALUATION USING THE USER'S POINT OF VIEWS

SEYED REZA MIRSALARI

DÉPARTEMENT DE GÉNIE INFORMATIQUE ET GÉNIE LOGICIEL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

THÈSE PRÉSENTÉE EN VUE DE L'OBTENTION

DU DIPLÔME DE PHILOSOPHIAE DOCTOR

(GÉNIE INFORMATIQUE)

OCTOBRE 2017

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Cette thèse intitulée:

A MODEL FOR SOFTWARE QUALITY EVALUATION USING THE USER'S POINT OF VIEWS

présentée par : MIRSALARI Seyed Reza

en vue de l'obtention du diplôme de : Philosophiae Doctor

a été dûment acceptée par le jury d'examen constitué de :

M. GUIBAULT François, Ph. D., président

M. ROBILLARD Pierre N., D. Sc., membre et directeur de recherche

M. KHOMH Foutse, Ph. D., membre

M. HARDY Simon, Ph. D., membre

DEDICATION

This thesis is dedicated to my wife and children,

Elham, Sonia, and Parsa

who have been a constant source

of support and encouragement.

I am truly thankful for having you in my life.

ACKNOWLEDGEMENTS

Firstly, I would like to express my sincere gratitude to my advisor Prof. Pierre N. Robillard for the continuous support of my Ph.D. study and related research, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my Ph.D. study.

Besides my advisor, I would like to thank the rest of my thesis committee: Prof. François Guibault, Prof. Foutse Khomh, and Prof. Simon Hardy, for their insightful comments and encouragement, but also for the hard questions, which incented me to widen my research from various perspectives. Many thanks to Prof. Michel C. Desmarais for his valuable inputs during the survey conduction.

My sincere thanks also go to the CTO and the employees at Technology Evaluation Canters, who provided me an opportunity to join their team as intern, and who gave access to the laboratory and research facilities. Without their precious support, it would not be possible to conduct this research.

Last but not the least, I would like to thank my family: my wife Elham and to my two children Sonia and Parsa for supporting me spiritually throughout writing this thesis and my life in general.

This thesis has been partially funded by Mitacs Canada project no. IT08408, IT04215, IT06440, and Technology Evaluation Centers.

RÉSUMÉ

Contexte: Dans le marché des logiciels en constante évolution, les acheteurs de logiciels sont confrontés à un défi majeur: parmi ces différents produits, lequel répond le mieux aux exigences et au budget des utilisateurs? Bien que la plupart des acheteurs de logiciels soient conscients de leurs besoins fonctionnels et budgétaires, les facteurs de qualité tels que la « disponibilité » ou la « fiabilité » ne sont généralement pas pris en compte. Les fournisseurs de logiciels parlent aussi rarement des aspects de qualité de leurs produits. Le défi principal est « comment susciter les attentes de qualité des utilisateurs ? », puis « comment déterminer les caractéristiques de qualité d'un produit logiciel ?». La comparaison de ces deux facteurs de qualité peut aider les acheteurs de logiciels à sélectionner le produit le mieux adapté et à ne pas gaspiller de budget supplémentaire pour des facteurs de qualité inutiles et ne pas acheter un produit qui ne couvre pas leurs exigences de qualité. Existe-t-il une méthode systématique pour rendre cette comparaison possible? Quels sont les facteurs qui influent sur la perception de l'utilisateur de la qualité du produit logiciel? **Objectif:** Dans cette thèse, nous visons à aborder la qualité du produit logiciel du point de vue des utilisateurs. L'objectif est de créer le profil de qualité attendu et observé du produit logiciel afin de démontrer les différences entre les qualités attendues par les utilisateurs et quelles qualités sont observées chez le produit logiciel. **Méthode:** Nous avons utilisé une stratégie empirique en utilisant une méthode basée sur un sondage pour créer le profil des caractéristiques de qualité attendues et observées. Après avoir développé un modèle de qualité standard, nous avons créé quatre types de questionnaires pour les utilisateurs finaux et les utilisateurs expérimentés, qui visent à susciter les facteurs de qualité. À titre d'étude de cas, nous avons mené trois enquêtes en deux phases dans l'industrie. Dans la phase I, nous avons demandé aux utilisateurs potentiels d'un produit logiciel en cours de développement, de répondre au questionnaire «qualité attendue». Dans la phase II, les utilisateurs d'un produit logiciel existant ont été invités à répondre au questionnaire «qualité observée». **Résultats:** Les résultats de la première phase montrent qu'il n'y a pas de différence significative entre les attentes de qualité de groupe d'utilisateurs final (end user) et d'utilisateurs expérimentés (power user). Dans la phase II, les résultats révèlent que les utilisateurs du département de développement, connus comme utilisateurs techniquement compétents, trouvent le logiciel plus performant que les utilisateurs d'autres départements de l'entreprise. **Conclusion:** Avec des profils de plan de qualité en main, il est possible d'effectuer un contrôle croisé utile entre les attentes de qualité spécifiques des utilisateurs et d'autres pilotes (exigences fonctionnelles et

architecture / conception), avant ou pendant le processus de développement de logiciels. Le contrôle croisé devrait viser à garantir qu'il existe suffisamment d'activités et de sous-activités dans le processus de développement de logiciels pour répondre aux attentes de qualité des utilisateurs. Sur la base des enquêtes menées, nous concluons que la qualité du logiciel du point de vue des utilisateurs dépend de la connaissance des utilisateurs sur les développements et la qualité de la technologie, en général, et sur le produit logiciel à l'étude, en particulier.

ABSTRACT

Context: In the ever-evolving software market, software buyers face a central challenge: Among these various products, which one best meets the users' requirements and budget? While most software buyers are aware of their functional and budgetary requirements, quality factors such as 'availability' or 'reliability' are not usually taken into account. Software vendors also rarely talk about the quality aspects of their products. The primary challenge is "how to elicit the users' quality expectations", and then "how to determine the quality characteristics of a software product". Comparing these two can assist the software buyers to select the best-fit product; not to waste extra budget for unnecessary quality factors, and not buy a product that does not cover their quality requirements. Is there any systematic method to make this comparison possible? What are the influencing factors that affect the user's perception of the software product quality? **Objective:** In this thesis, we aim to address the quality of the software product from the users' point of view. The goal is to create the expected and observed quality profile of the software product to demonstrate the differences between what qualities were expected from the users' side, and what qualities are observed in the software product. **Method:** We employed an empirical strategy using a survey-based method to create the profile of expected and observed quality characteristics. After developing a standard-based quality model, we created four types of questionnaires for end users and power users, which aim to elicit the quality factors. As a case study, we conducted three surveys in two phases in the industry. In phase I, we asked the potential users of a software product which was under development, to answer the 'expected-quality' questionnaire. In phase II, the users of an existing software product were asked to answer the 'observed-quality' questionnaire. **Results:** The results of the first phase show that there is no significant difference between the quality expectations of the end and power user groups. In phase II, the results reveal that the users in development department who are known as technically knowledgeable users find the software as higher quality than the users in other departments of the company. **Conclusion:** With quality plan profiles in hand, it is possible to perform a useful crosscheck between users' specific quality expectations and other drivers (functional and architecture/design requirements), before or during the software development process. The crosscheck should be aimed to guarantee that there are enough activities and sub-activities in the software development process to support the users' quality expectations. Based on the conducted surveys, we conclude that the software quality from

the users' point of view depends on the knowledge of the users about the software developments and quality, in general, and on the software product under study, specifically.

TABLE OF CONTENTS

DEDICATION III

ACKNOWLEDGEMENTS IV

RÉSUMÉ V

ABSTRACT VII

TABLE OF CONTENTS IX

LIST OF TABLES XIII

LIST OF FIGURES XVII

LIST OF APPENDICES XIX

CHAPTER 1 INTRODUCTION 1

1.1 Software quality and software quality evaluation 2

1.2 Systematic scoping review 4

1.2.1 The goal of the scoping review 5

1.2.2 Identifying relevant studies 5

1.2.3 Select studies to include 6

1.2.4 Extract and charting data from included studies 6

1.2.5 Collate, summarize and report results 8

1.2.6 Conclusion 17

1.2.7 The position of the current thesis in the scope 18

1.3 Models, norms, and standards in software quality domain 19

1.4 Data collection for quality evaluation 21

1.5 Quality Profile: Expected vs. Observed quality 22

1.6 Research Approach 25

1.7 Objective of the Research 25
1.8 Structure of the thesis 25
CHAPTER 2 LITERATURE REVIEW 27
2.1 The difficulties in surveys for software quality evaluation 27
2.1.1 Goal 27
2.1.2 Data extraction: The targeted context values 27
2.1.3 Strategy 27
2.1.4 Search string 28
2.1.5 Libraries 28
2.1.6 The retrieved papers 28
2.1.7 The reported difficulties 38
2.1.8 Conclusion 40
2.2 The participants' knowledge 41
2.2.1 Goal 41
2.2.2 Data extraction: The targeted context values 41
2.2.3 Strategy 42
2.2.4 Search string 42
2.2.5 Libraries 42
2.2.6 The retrieved papers 42
2.2.7 Conclusion 50
2.3 Conclusion of the literature review 50
2.4 Research motivation 51
CHAPTER 3 METHODOLOGY 52
3.1 Our quality evaluation model 52

3.2 Questionnaire creation....53
3.3 Questionnaire refinement....53
3.4 Association with the standard: A coding activity....56
3.5 The case studies....58
3.5.1 Case #1:....59
3.5.2 Case #2:....59
CHAPTER 4 SYNTHESIS OF PUBLICATIONS....61
4.1 Article 1: Industrial Validation of an Approach to Measure Software Quality....61
4.1.1 The results....62
4.2 Article 2: Expected Software Quality Profile: A Methodology and a Case Study....65
4.2.1 The results....66
4.3 Article 3: Does Participants' Knowledge Affect the Survey Results? A Systematic Literature Review in Software Engineering Domain....71
4.3.1 The results....71
CHAPTER 5 GENERAL DISCUSSION....81
5.1 Systematic outlier removal....81
5.2 Conclusion....85
5.3 Where the knowledge lacks....85
5.4 The indicators....86
5.5 Question Applicability vs. don't know....90
5.6 The knowledge missing items....92
5.7 Correspondance with the reality....93
5.8 Threats to validity....94
5.9 Conclusion....96
CHAPTER 6 CONCLUSION AND RECOMMENDATIONS....98

BIBLIOGRAPHY 101

APPENDICES 110

LIST OF TABLES

Table 1: Number of articles from electronic databases 6

Table 2: The selected papers for mapping study 8

Table 3: The aspect of each retrieved paper 10

Table 4 : Questionnaire / Interview per Paper 11

Table 5: The data capturing methods addressed in each paper 12

Table 6: Data analysis method in each paper 14

Table 7 : Compiling the papers to extract their goal and outcomes 16

Table 8: Number of retrieved papers for LR 29

Table 9 : List of 27 retrieved papers for the 1st phase of the LR 30

Table 10: List of 11 retrieved papers for 2nd phase of the LR 43

Table 11: Rating Scales 53

Table 12: Sample questions and rating scales 55

Table 13: Final Questions – sample 56

Table 14: Elements of a hypothetical Model matrix 57

Table 15: The Likert scales 60

Table 16: Number of comments per expert 62

Table 17 : Feedback Types 63

Table 18: Number of identical feedbacks per Experts 64

Table 19: Element of a hypothetical Value Matrix 66

Table 20: Numbers and percentage of each rating scales: End user questionnaire – Quality in-use 67

Table 21: The p-value resulting from Fisher's test 68

Table 22: The scales and their normalized values for Effectiveness in the 2nd questionnaire 73

Table 23: p-values obtained from Fisher's Test for three sample quality characteristics of both questionnaires 76

Table 24: the Quality Characteristics and the p-values 76

Table 25: Number of X's for each question in the both questionnaires 77

Table 26: The percentage of Xs received for each question in both questionnaires. The * denotes the outliers 82

Table 27: The identified outlier questions based on the given thresholds 83

Table 28 : Justifications for outliers 84

Table 29: SharePoint Involvement Percentage (SPIP) 87

Table 30: Association of four sample questions with the departments. Y=Associated N=Not 88

Table 31: Applicability of the questions for the departments 89

Table 32: Involved Departments 90

Table 33: The Applicability-Data Matrix 91

Table 34: The questions with the number of 'X Yes' values categorized by 'Knowledge missing' and 'don't care' labels 93

Table 35: Tech-savvy departments at TEC 94

Table 36: SharePoint Involvement vs. Questions Involvement 95

Table 37 : Rating scales 113

Table 38: An example of weight determination 114

Table 39: Number of comments per expert 115

Table 40 : Feedback Types 115

Table 41: Number of identical feedbacks 116

Table 42: Rating scales 123

Table 43: Sample questions and rating scales 123

Table 44: Sample Final Questions 124

Table 45: Elements of a hypothetical Model matrix 124

Table 46: Element of a hypothetical Value matrix 126

Table 47: Numbers and percentage of each rating scales: End user questionnaire – Quality in-use 126

Table 48: The p-value resulting from Fisher's test 126

Table 49: Search string for SLR 137

Table 50: Details of paper selection and elimination in the SLR 138

Table 51 : Descriptions: Paper's context values 142

Table 52: The papers that report that participants' knowledge AFFECTS the survey results 145

Table 53: The papers that report that there is NO effect of participants' knowledge on survey results 146

Table 54 : The papers that do not care (NEUTRAL) about the effect of participants' knowledge on survey results 147

Table 55 : The Likert scales 151

Table 56: The scales and their normalized values for Effectiveness in the 2nd questionnaire 153

Table 57: p-values obtained from Fisher's Test for three sample quality characteristics of both questionnaires 155

Table 58: the Quality Characteristics and the p-values 156

Table 59: Number of X's for each question in the both questionnaires – the reds are the outliers 157

Table 60: The percentage of X's received for each question in both questionnaires. The * denotes the outliers 159

Table 61: The identified outlier questions based on the given thresholds 160

Table 62 : Justifications for outliers 162

Table 63: SharePoint Involvement Percentage (SPIP) 164

Table 64: Association of four sample questions with the departments 165

Table 65: Applicability of the questions for the departments 166

Table 66 : Involved Departments 167

Table 67: The Applicability-Data Matrix 168

Table 68 : The questions with the number of 'X Yes' values categorized by 'Knowledge missing' and 'don't care' labels 169

Table 69 :Tech-savvy departments at TEC 171

Table 70: SharePoint Involvement vs. Questions Involvement 172

LIST OF FIGURES

Figure 1: Retrieving the articles from electronic databases 7

Figure 2: Current state of the quality evaluation research in software engineering field 9

Figure 3: Data Capturing methods 13

Figure 4: The papers and the reported data analysis methods 14

Figure 5: Presentation Methods 15

Figure 6: The position of the thesis on the scope: the orange color pieces 19

Figure 7: Method to build quality deviation artifact; iterative and incermental 24

Figure 8: Research process to create the Quality Profile 52

Figure 9: Number of questions associated with quality in-use characteristics for end users 57

Figure 10: Questions and the number of associated quality characteristics 58

Figure 11: The average % of each received feedback type 63

Figure 12: Number of questions for each decision 65

Figure 13: Expected quality profile - End users - Quality in-use 69

Figure 14: Expected Quality profile – End users - Product quality 69

Figure 15: Pareto analysis for “Satisfaction” – End users 70

Figure 16: The scales in both surveys 72

Figure 17: Quality Profiles for 'Supported' scale 73

Figure 18: Quality Profiles for ‘So-So’ scale 74

Figure 19: Quality Profiles for ‘Not Supported’ scale 74

Figure 20: Quality Profiles for ‘Don’t know’ answers 75

Figure 21: Method to build quality deviation artifact 112

Figure 22: Software Quality Profile diagrams (QPD) for the planned quality and the observed quality (Actual) and quality deviation diagram between the two profile diagrams. 112

Figure 23: Quality measurement model based on ISO 25021 114

Figure 24: The average percentage of each received feedback type.....115

Figure 25: Number of questions for each decision 116

Figure 26: Standard Product Quality Model – ISO/IEC 25000 122

Figure 27: Number of questions associated with quality in-use characteristics for end users.....125

Figure 28: Number of quality characterisitics related to each question – end user's questionnaire 125

Figure 29: Expected quality profile - End users - Quality in-use.....127

Figure 30: Expected Quality profile – End users - Product quality 127

Figure 31: Pareto analysis for "Satisfaction" – End users 127

Figure 32: Growth of questionnaire-based surveys in SE since 2007.....135

Figure 33: Research process to create the Quality Profile 149

Figure 34: The scales in both surveys 152

Figure 35: Quality Profiles for 'Supported' scale 153

Figure 36: Quality Profiles for 'So-So' scale.....154

Figure 37: Quality Profiles for 'Not Supported' scale 154

Figure 38: Quality Profiles for 'Don't know' answers 154

LIST OF APPENDICES

Appendix A – ARTICLE 1: Industrial Validation of an Approach to Measure Software Quality 110

Appendix B – ARTICLE 2: Expected Software Quality Profile: A methodology and a case study 121

Appendix C – ARTICLE 3: Does Participants' knowledge affect the survey results? A systematic literature review in software engineering domain.....132

Appendix D – Questionnaire: Software Quality Profile – End users.....183

Appendix E – Questionnaire: Software Quality Profile – Power users 188

CHAPTER 1 INTRODUCTION

On time delivery of a software product to the customer, within budget, and in a correct functionality form, does not guarantee that the software will be well accepted by the targeted users. The software source code may be difficult to modify, and it may lead to high costs of maintenance. The software may be unnecessarily machine dependent or hard to integrate with other organizational systems. A high-quality software is defined as what it provides value and avoids potential negative effects for the costumers. This requires the software to cover a wide spectrum of quality factors. However, because the quality factors are usually expressed in a qualitative form by the users, identifying and evaluating the software-specific quality factors form the users' point of view is always challenging [1].

In this thesis, we have taken the subject of *software quality evaluation* into consideration. For this purpose, a scoping review has been performed to examine the range of research activities and to analyze the trends in the area of 'software quality evaluation' field, and to summarize and categorize the research findings. This goal is achieved by emphasizing on the *aspects* and the *steps* of the quality evaluation process that have been applied by the researchers.

One of the primitive steps of a quality evaluation process is to select the most appropriate quality model, norm, or standard. Experienced and academic expert have defined software quality standards decades ago. Standards bring positive advantages for enterprises when they need to acquire a software, or evaluate the quality of an existing software application. Although each quality model or quality standard represent its own specific quality characteristics and evaluation method, it is required for all of them to collect valid data correctly and efficiently. The literature emphasizes that collecting the right data items is necessary for quality evaluation process[2]. The quality evaluation process is easier when the source of the appropriate date and how to retrieve them are known.

In this thesis, we describe how we used the correct data to formulate our quality evaluation methodology and to perform our case studies. The next chapters explain how to create the Quality Plan Profile by eliciting the expected quality based on customer quality requirements, and to create the Product Quality Profile by quantifying the quality characteristics of the software products. In the last chapter, we briefly explain how the Quality Deviation Artifact can help the practitioners to evaluate the differences between expected quality and obtained quality.

This thesis presents the literature review that stresses the difficulties in conducting the surveys in software engineering field, and the effects of the participants' knowledge on the results of the surveys. The following chapters present the methodological approaches used in our research. At the end, the results are discussed and the lessons learned from our case studies are presented.

In the following sections, the above-mentioned topics are described in detail.

1.1 Software quality and software quality evaluation

Since this thesis aims to address the software quality, it would be appropriate if we start with the definition of the term "quality". However, the academia, as well as the industry, have not, and may never, come to a single definition of the term quality. For example, the ISO/IEC Systems and Software Engineering—Vocabulary (ISO/IEC/IEEE 24765:2010) [3] has the following set of definitions for "quality":

1. "The degree to which a system, component, or process meets specified requirements."
2. "Ability of a product, service, system, component, or process to meet customer or user needs, expectations, or requirements."
3. "The totality of characteristics of an entity that bear on its ability to satisfy stated and implied needs."
4. "Conformity to user expectations, conformity to user requirements, customer satisfaction, reliability, and level of defects present."
5. "The degree to which a set of inherent characteristics fulfills requirements."

The definitions of software quality, held by the founders of modern quality assurance, Philip B. Crosby, and Joseph M. Juran reflect a different conception of software quality: "Quality means conformance to requirements" [4]. From the Crosby's point of view, "quality consists of those product features which meet the needs of customers and thereby provide product satisfaction". Juran [5] also believes that "quality consists of freedom from deficiencies" [5].

Almost in all definitions of quality, it is emphasized to achieve customer satisfaction and to view the fulfillment of customers' real needs as the true goal of software quality. Although the term "quality" is widely used in various domains such as philosophy, economics, marketing, operations

management, etc., Schulmeyer concluded, “Quality is a complex and multifaceted concept.” [6]. Schulmeyer describes quality from five different perspectives:

- **Transcendental perspective:** the quality can be seen as something that can be perceived but not necessarily defined. What one customer considers good software quality may not be high enough quality for another customer.
- **Manufacturing perspective:** this perspective can be interpreted as conformance to the specifications. This perspective refers to the ability to produce a product to pre-defined specification over and over within accepted tolerances.
- **User perspective:** Quality can be seen as the robustness of the product or service for use. There are many stories told by software practitioners related to the software products that met their specifications but did not function adequately when deployed into operations. This reveals the importance of ‘context of use’. This perspective of quality not only considers the viewpoints of the individual users but also their context of use. For example, what a novice user might consider a “quality” user interface might drive a power user to distraction with pop-up help and warning messages that require responses.
- **Product perspective:** in the literature, the quality characteristics or the quality attributes, are also called the “ilities” of the software product, such as reliability, usability, availability, flexibility, maintainability, and portability. However, the quality factors do not all end in “ility.”, correctness, fault tolerance, integrity, efficiency, security, and safety are also examples of quality attributes. The more the software has high levels of these characteristics, the higher its quality is considered to be.
- **Value-based perspective:** How much is the customer willing to pay for Quality? This perspective reveals the notion of “good enough” software quality. Are people willing to pay as much for high-quality video game software as they are for high-quality software in biomedical devices or the high-quality software for airplane navigation systems?

Considering these varieties in the definition and the perspectives of software quality, evaluating the quality is also challenging. Evaluating the various aspects of software quality is considered to be an effective tool for the support of control activities and the initiation of process improvements during the development and the maintenance phases. These measurements apply to the functional

quality, productivity, and organizational aspects of the project. Among the software quality metrics, we can list metrics for:

- Quality of software development and maintenance activities
- Development teams' productivity
- Helpdesk and maintenance teams' productivity
- Software faults density
- Schedule deviations
- User satisfaction

Since the quality factors are usually presented in a qualitative manner, evaluating the quality factors needs to apply appropriate data capturing methods and analyzing techniques. In the literature, several quality evaluation models have been introduced. In the next section, we briefly explain the main models and standards that address the quality evaluation notion.

1.2 Systematic scoping review

A scoping study is neither to address very specific research questions nor, consequently, to assess the quality of included studies. It tends to address broader topics where many different study designs might be applicable [7]. The Systematic Scoping Review (SSR) is defined as "a form of knowledge synthesis that addresses an exploratory research question aimed at mapping key concepts, types of evidence, and gaps in research related to a defined area or field by systematically searching, selecting and synthesizing existing knowledge" [8]. According to this definition, we set the objective for our scoping review to examine the range of research activities and analyzing the trends in the area of 'software quality evaluation' field, and to summarize and categorize the research findings.

The steps for performing a comprehensive scoping review has been addressed in the literature. In our scoping review, we follow the steps which are discussed in [7]. They are:

1. Identify the research goal
2. Identify relevant studies
3. Extract & charting data from included studies

4. Collate, summarize & report results

In the sections below, we explain how these steps have been conducted in this scoping review.

1.2.1 The goal of the scoping review

As it is mentioned earlier, the goal of this scoping review is to understand how the researchers in software engineering field have addressed the quality evaluation subject. For this purpose, we emphasized on the aspects and the steps of the quality evaluation process that have been used by the researchers.

1.2.1.1 Quality Evaluation Aspects

The ISO/IEC 25010 standard looks at the product quality from two viewpoints: "quality in-use" model which is composed of five characteristics (Effectiveness, Efficiency, Satisfaction, Freedom from Risk, Context Coverage) that relate to the outcome of interaction when a product is used in a particular context of use, and "product quality" model which is composed of eight characteristics (Functional Suitability, Performance Efficiency, Compatibility, Usability, Reliability, Security, Maintainability, Portability) that relate to static and dynamic properties of the software product. Some of the characteristics are further decomposed into sub-characteristics [9]. According to what identified in ISO/IEC 25010, we set the goal of our SSR to categorize the studies based on the two standard aspects: quality in-use and product quality.

1.2.1.2 Quality Evaluation Steps

In ISO/IEC 25040 a comprehensive list for software quality evaluation is presented. We summarized the list in three main steps: 1) data capturing for quality evaluation, 2) analyzing the captured data, and 3) presenting the evaluation results. In this scoping review, we planned to review the reports and categorize them based on the above steps.

1.2.2 Identifying relevant studies

For retrieving the related studies for our systematic scoping review, we used five electronic databases: Compendex, Inspec, ACM Digital Library, IEEE Xplore, and Web of Science. A general search string was used to extract the most SSpossible related articles: *"Software Engineering"*

AND "Quality Evaluation". The search string was used to search in keywords, titles, and abstracts fields.

The number of retrieved articles from each database is presented in Table 1.

Table 1: Number of articles from electronic databases

Database	# of Papers
Compendex	205
Inspec	93
ACM Digital Library	30
IEEE Xplore	140
Web of Science	40

1.2.3 Select studies to include

Using our search string, we retrieved 508 records from the five databases. These databases cover a wide range of subjects in software engineering literature; we exclude the items that did not provide useful information for our review. In the next section, we explain our elimination procedure.

1.2.4 Extract and charting data from included studies

For selecting the most appropriate papers, it is needed to distinguish and eliminate the unrelated ones. The data in Figure 1 shows the steps and the results of the paper selection process. The steps are sequentially presented as below:

1. Removing the duplicates: 86 items were excluded.
2. Removing non-scientific papers: conference proceeding, announcements, posters, and non-peer reviewed scientific studies, books, and book chapters: 8 papers were excluded.
3. Removing non-English papers: 16 papers were excluded.
4. Removing before 2010 publications: we limited our review to the papers published since 2010 to 2017. 253 papers were excluded.
5. Removing based on Title: the papers based on non-related titles were removed. 50 papers were excluded.
6. Removing base on the abstract and the conclusion: reviewing the abstract and the conclusion of the remained papers resulted in excluding 30 papers.

7. Removing not-available PDF: we also excluded 8 papers that the full PDF were not accessible.
8. Removing the Theoretical papers: we limited our review only to the papers that reported an 'empirical' study. Therefore we excluded the theoretical, academic, and any other types which are 33 papers.
9. Removing the short papers: to make our review more valid, we kept the papers with five pages and more. This step resulted in excluding 9 more papers.

After performing the above elimination steps, 13 articles remained. The steps are illustrated in Figure 1.

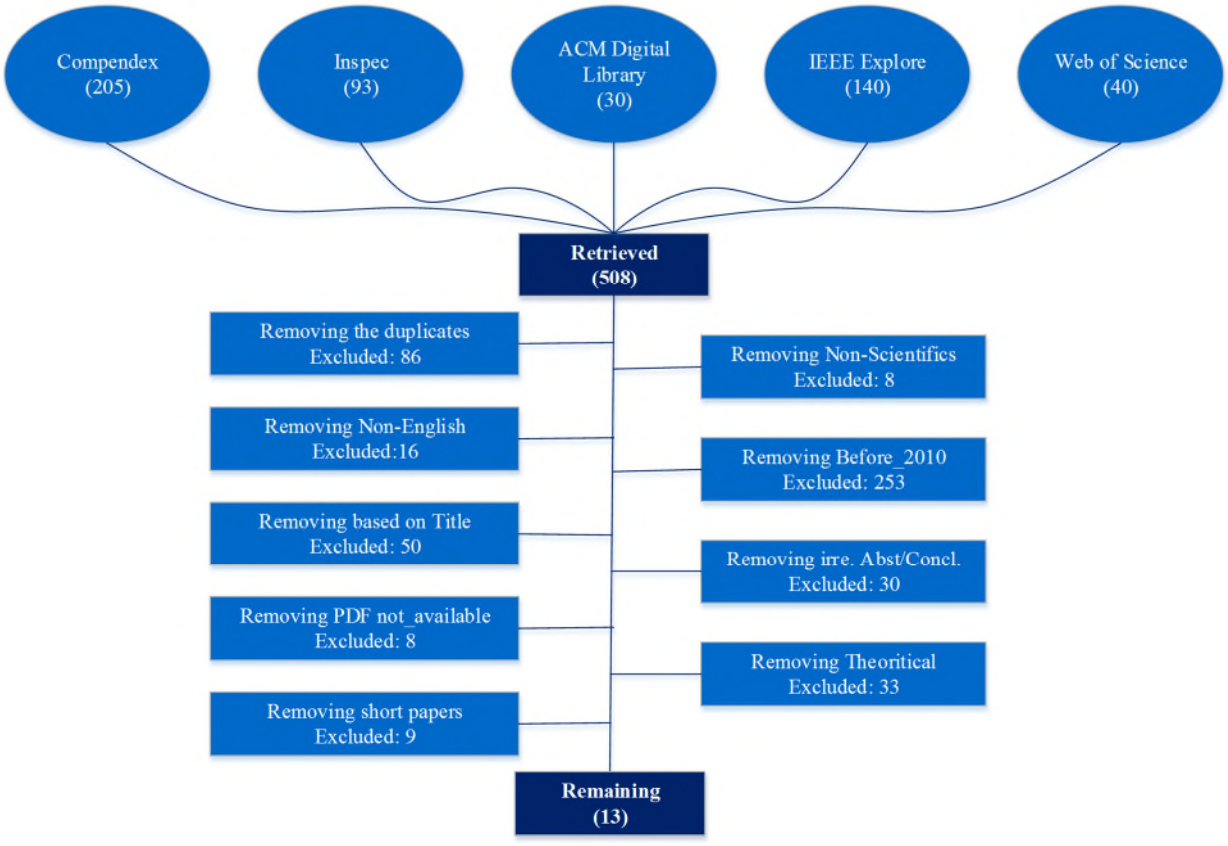


Figure 1: Retrieving the articles from electronic databases

The remained papers are listed in Table 2. The papers are identified as P_1 to P_{13} as well as the bibliographical references.

Table 2: The selected papers for mapping study

ID	Titles
P_1	Adapting Software Quality Models: Practical Challenges, Approach, and First Empirical Results [10]
P_2	An empirical evaluation of the quality of interoperability specifications for the web [11]
P_3	An experiment of software quality evaluation in the audio-visual media preservation context [12]
P_4	Analyzing the reliability of open source software projects [13]
P_5	Are Comprehensive Quality Models Necessary for Evaluating Software Quality? [14]
P_6	Behavioral economics in software quality engineering [15]
P_7	Better Code for Better Apps: A Study on Source Code Quality and Market Success of Android Applications [16]
P_8	Empirical validation of website quality using statistical and machine learning methods [17]
P_9	Evaluation of academic website using ISO/IEC 9126 [18]
P_{10}	Expected software quality profile: A methodology and a case study [19]
P_{11}	Hybrid functional link artificial neural network approach for predicting maintainability of object-oriented software [20]
P_{12}	Quality evaluation of conceptual level object-oriented multidimensional data model [21]
P_{13}	XML Schema metrics for quality evaluation [22]

1.2.5 Collate, summarize and report results

In this section, we explain the findings after reviewing the 13 papers. As it was discussed earlier, we retrieve the data in two categories: aspects and steps. Figure 2 illustrates a summary of what we extract from scoping review process. It shows the current state of the quality evaluation research in software engineering field in terms of aspects and steps. in the following sections we explain the results in more details.

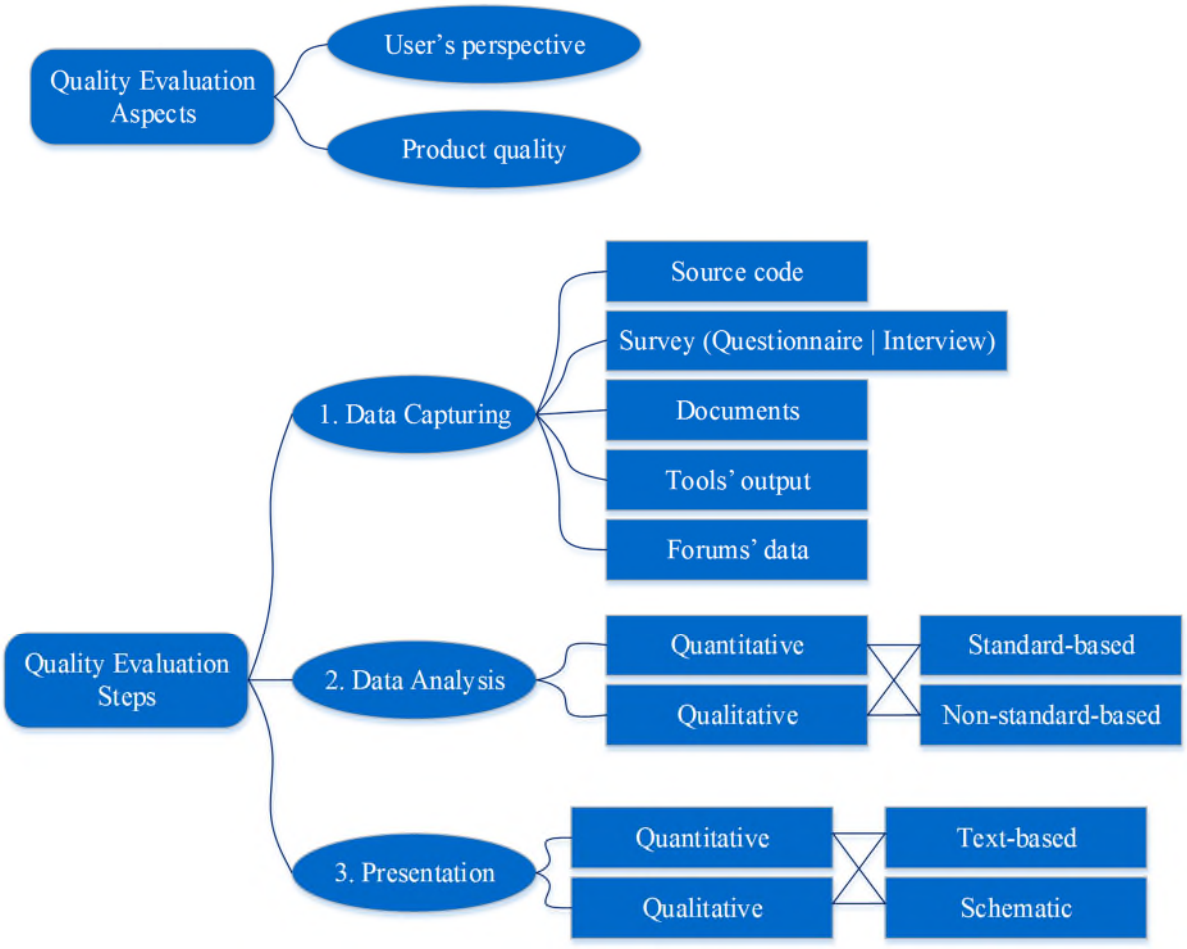


Figure 2: Current state of the quality evaluation research in software engineering field

1.2.5.1 Quality evaluation aspects

Based on the quality model presented in ISO/IEC 25010, the software product quality can be viewed from the 'quality in-use' and from the 'product quality' aspects. The data in Table 3 shows the aspect of the papers that we found during our review. The coding procedure is performed based on the definitions in ISO/IEC 25010. If a paper addresses a quality characteristic or sub-characteristic related to Quality in-use, we coded as Quality in-use and the same way for Product Quality.

Table 3: The aspect of each retrieved paper

id	Quality in-use	Product quality
P_1	X	X
P_2		X
P_3	X	X
P_4		X
P_5		X
P_6		X
P_7	X	X
P_8		X
P_9		X
P_{10}	X	X
P_{11}		X
P_{12}		X
P_{13}	X	X

The data in Table 3 shows that there is a gap in addressing the quality in-use aspect in the literature for quality evaluation. All the retrieved papers are addressing the product quality aspect, while few of them address also the quality in-use.

1.2.5.2 Quality evaluation steps

We divided the steps of the quality evaluation in three parts: Data Capturing, Data Analysis, and Reporting. Repeatedly, we reviewed the papers and performed the coding procedure. In the following section, we explain the results that we obtain in each step.

1.2.5.2.1 Data capturing

In this step, we aimed to find out how the researchers capture data for the quality evaluation process. We look for the methods that the researchers employed to retrieve the data for quality measure elements. We found that these data are being retrieved by reviewing the:

1. **Source code:** Analyzing the source code of the software application under review is one the methods that is used by a number of the researchers. The reports that reviews source code to capture data are: P_1, P_2, P_4, P_5, P_7, P_8, P_{11}, P_{12}, and P_{13}.
2. **Surveys:** In many cases, the authors reported that they asked the stakeholders to participate in a survey, or in an interview. The data in Table 4 shows which paper is using questionnaire or interview for data capturing. It shows that most researchers used the Questionnaire as a reliable tool for collecting data.

Table 4 : Questionnaire / Interview per Paper

id	Questionnaire	Interview
P_1	X	
P_2		
P_3	X	X
P_4		
P_5		
P_6	X	
P_7	X	
P_8		
P_9	X	
P_{10}	X	
P_{11}		
P_{12}	X	
P_{13}		X

3. **Documents**: Reviewing the documents is one of the methods that the researchers use to capture data. Our SSR shows that two papers used the documents for quality evaluation analysis. In the P_2 the results of a readability test are presented. In P_4 the authors presented a quality evaluation model that analyses the documentations in addition to other aspects.
4. **Tools or repositories:** in some reports, we found out that the researchers are using the output of tools or the data stored in the repositories to evaluate a software product. For example, three tools are taken into consideration for collecting data in P_4 in a case study. The tools are Sourceforge, Freshmeat, and Openhub. In P_3 an experiment of software quality evaluation, for storage tools in the audio-video preservation environment is reported.
5. **Forums:** using the data of the forums is another method for quality evaluation. For example, in P_4 the data of the official forums are used as the resource for quality evaluation.

Table 5 presents the methods that researchers are using for quality evaluation purpose.

Table 5: The data capturing methods addressed in each paper

Id	Source code	Survey	Document	Tools	Forums
P_1	X	X			
P_2	X		X		
P_3		X		X	
P_4	X		X	X	X
P_5	X				
P_6		X			
P_7	X	X			
P_8	X				
P_9		X			
P_{10}		X			
P_{11}	X				
P_{12}	X	X			
P_{13}	X	X			

In Figure 3 the results of the data capturing step are presented. It shows that the majority of the papers have used a ‘survey’ for data capturing. Reviewing the ‘source code’ is also popular among the researchers. Each rectangle represents a data capturing method. The data in the figure are the papers IDs. For example, for the paper P_4 the researchers have used the source code, tools, documents and forums’ data for evaluating the software quality.

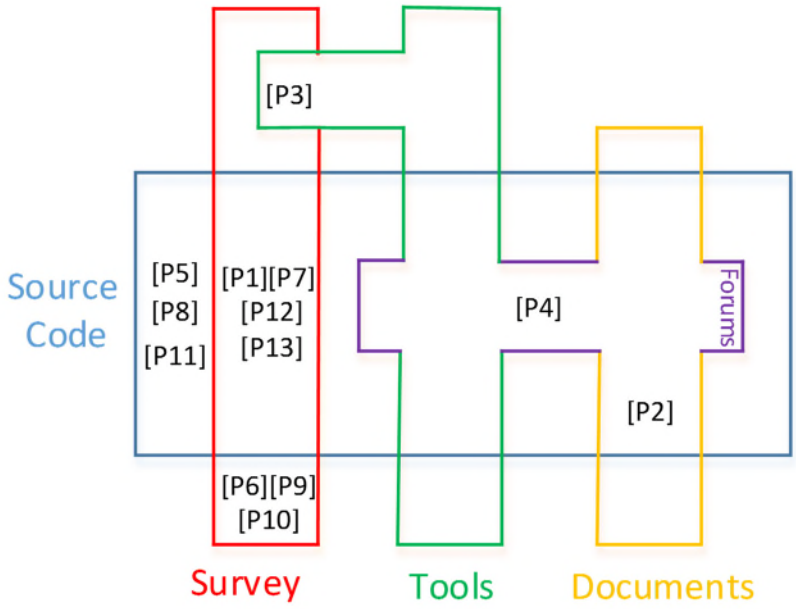


Figure 3: Data Capturing methods

1.2.5.2.2 Data analysis

The researchers use various approaches for analyzing the captured data. Because of the diversity of the methods, we categorized them in two high level approaches:

1. Quantitative data analysis: If a study uses mathematical or statistical modeling to analyze the captured data, we labeled that as a quantitative analysis study. In most cases in quantitative studies, absolute scales are being used for analyzing the data.
2. Qualitative data analysis. If a study uses unquantifiable information using ordinal scales such as Likert scales, we consider it as a qualitative study.

As illustrated in Figure 2, in both quantitative and qualitative methods, it is possible to use standard-based and nonstandard-based methods. If a study employs a quality evaluation model according to an ISO or IEEE standard, or according to a local norm, we consider it as a standard-based study. Otherwise, it is considered as a nonstandard-based one. Table 6 shows the retrieved data analysis methods and the papers of each method. It shows that there are 12 papers that analyze the data by using a standard and quantitative approach.

Table 6: Data analysis method in each paper

id	Quantitative	Qualitative	Standard-based	non-Standard based
P_1	X	X	X	
P_2	X		X	
P_3	X	X	X	
P_4	X		X	
P_5	X		X	
P_6	X		X	
P_7	X		X	
P_8	X			X
P_9	X		X	
P_{10}	X	X	X	
P_{11}	X		X	
P_{12}	X		X	
P_{13}	X	X	X	

In Figure 4 the results of the data analysis step are presented. It shows that the majority of the papers have used the quantitative and standard-based methods. The data in Figure 4 are the papers IDs. Each rectangle represents method while the green cloud shape includes the papers shared between quantitative, qualitative, and standard-based methods.

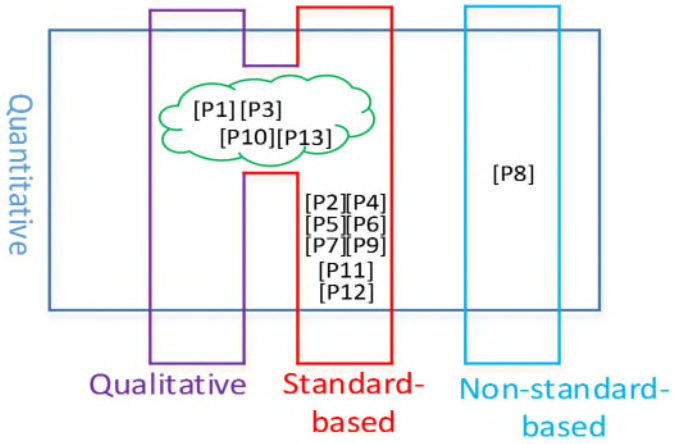


Figure 4: The papers and the reported data analysis methods

1.2.5.2.3 Reporting

For presenting and reporting the results of the quality evaluation, researchers use various methods. From a high level of abstraction, as illustrated in Figure 2, depends on the nature of the results, in terms of quantitative or qualitative metrics used during the data gathering and data analysis, researchers use quantitative and qualitative methods to demonstrate the evaluation results. In more details, we identify the studies presenting the results in a text-based manner, such as a table of metrics and values, or in a schematic and illustrative manners such as diagrams, charts, or figures. Figure 5 shows the presentation methods and the number of papers of each method. It shows that 11 papers have used a text based or schematic method quantitatively.

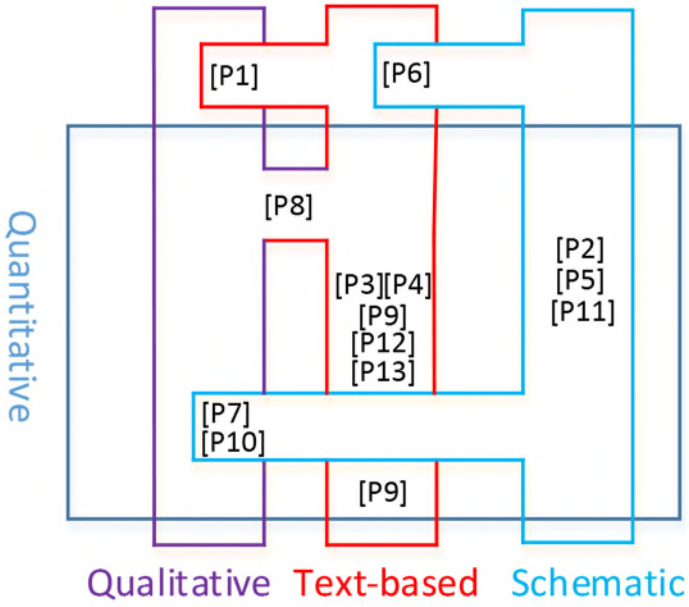


Figure 5: Presentation Methods

1.2.5.2.4 Goal and Outcomes

We compiled the research objective and the outcome of the retrieved papers to check how, and in what extent, the researchers have employed the quality models and evaluation tools. The data in Table 7 presents the result of this compilation.

Table 7 : Compiling the papers to extract their goal and outcomes

id	Goal and Outcomes of the paper
P_1	The authors presented a tool-supported approach for efficient adaption of quality models. The quality models obtained by this approach are more adapted than those obtained following an ad-hoc approach.
P_2	This study verifies the applicability of the automatic quality evaluation. The paper evaluates the results of the proposed quality evaluation, elaborates on the extent and scope of the available means, and discusses the applicability of the used quality evaluation system. The authors conclude that an automated quality evaluation system can identify weak points of web-based semantic interoperability specifications by measuring and scoring violations against the policies.
P_3	In this paper, a software quality model customizable for the audio-visual context has been performed. The authors highlighted the most important points: finding an agreement among stakeholder, groups for specializing the quality model, defining a common understanding of the characteristics and sub-characteristics, better expressing the requirements to avoid generic and non-quantifiable terms i.e. “some”, “optimal” “short”, etc., specializing the measurement plans starting from the suggestions of the international standards so to reduce effort due too much generic and abstract quality evaluations.
P_4	This paper evolves a quality model for Free/Open Source Software projects, for including reliability aspects. The authors concluded that it is important not only to consider the quality of the software, but also other distinctive features of the open source projects.
P_5	This paper investigates if it is possible to build a focused quality model with similar evaluation results as a comprehensive quality model but with far less measures needed to be collected and reduced effort. The authors concluded that they can build focused quality models to get an impression of a system's quality similar to comprehensive models.
P_6	This article proposes a simplified method to manipulate the observed. The proposed experiment has been conducted among professional software evaluators. The results show the significant negative influence of negative experience of users on final opinion about software quality regardless of its actual level.
P_7	The authors investigated the contribution of the code quality in the market success of Android apps in the Google Play store. They determined whether there is a relationship between product quality and market success. The result show that the quality of the source code has a marginal impact into the indices that describe the market success.
P_8	The paper, computes 22 metrics using a Matlab tool. Website quality prediction is developed using statistical and some machine learning methods. The results show that hat quality models have a significant relevance with design metrics and the machine learning methods have a comparable performance with statistical methods.
P_9	In this research, academic website quality evaluation is conducted using ISO 9126 in Telkom University website. This evaluation is conducted to ascertain whether there is any characteristics of the website that need to be improved. Based on the evaluation results the authors obtained three characteristics that need to be improved, they are reliability, usability and functionality.
P_{10}	The authors present a methodology to create the expected quality profile.

Table 7 : Compiling the papers to extract their goal and outcomes

id	Goal and Outcomes of the paper
P_{11}	In this paper, three artificial intelligence techniques are applied to design a model for predicting maintainability. The results show that feature reduction techniques are very effective in obtaining better results while using FLANN-Genetic.
P_{12}	In this paper, a set of quality metrics have been proposed along with the theoretical validation for quality measurement of conceptual level object-oriented multidimensional data model. The paper also describes the designer level and user level viewpoints of quality evaluation through the criteria like complexity, completeness, expressiveness and analyzability. Finally, the work focuses on the empirical validation of the set of metrics and measurements. The empirical validation process shows the usefulness of the proposed metrics for the assessment of operability factor of conceptual level multidimensional data model.
P_{13}	This paper proposes a quality measuring approach, based on existing software engineering metrics, additionally defining the quality aspects of XML Schemas. The results illustrate the influence of XML Schema's characteristics on its quality and evaluate the applicability of metrics in the measurement process, a useful tool for software developers while building or adopting XML Schemas.

The data presented in Table 7, shows that the researchers have used the quality evaluation methods and tools for various objectives, and they obtained various outcomes.

1.2.6 Conclusion

In this scoping review, we employed a systematic approach understanding better how the researchers in the software engineering domain are capturing and analyzing data and how they report the results for software quality evaluation purpose. We started to get a set of software quality evaluation papers together, then through an elimination procedure, we came up with 13 empirical reports. The categorizations presented in ISO/IEC 25000 helped us to the make group and analyze our retrieved data.

Our findings showed that in all the selected papers the researchers performed their evaluation process from 'product quality' point of view. The results also showed that the researchers are mostly interested in capturing data by using the source code analysis and conducting the surveys. After capturing data, the researchers are using quantitative and standard-based methods for analyzing the captured data. For presenting the results, the researchers are mostly using quantitative and text-based presentations.

Our results also showed the gaps in the domain. The gaps are the data resources and the methods that rarely used by the researchers in the field. Addressing the quality in-use, using Forums' data,

using qualitative analysis methods, presenting data in a schematic manner are the items that lack in the reports. These items can be considered as the research avenues for future work in software quality evaluation.

1.2.7 The position of the current thesis in the scope

In this thesis, we addressed the quality evaluation from both aspects: quality in-use and product quality. For data capturing we conducted the surveys and the questionnaire as the instrument. We also had a quantitative and standard based approach for our data analysis. At the end, we presented the results using a quantitative approach. The orange shapes in Figure 6 schematically represent the pieces of the scope that have been touched within the current thesis. Since previously we have presented the quality evaluation methods, tools, research objectives, and outcomes in the literature, we have selected a combination of the elements that guarantees the novelty and originality of our thesis. Our goal in this stage was to obtain the results different than what have been obtained by previous researchers.

In Figure 6, the values in the parentheses represent the percentage that each item takes in comparison with other items in the same category. For example, 62% of the papers have used the ‘survey’. During the coding and labeling process, some papers were associated with multiple data capturing, analysis, or presentation methods. For example, the paper P1 claims that the researchers have used both ‘source code’ and ‘survey’ for data capturing. Because of these overlaps, the percentages in Figure 6 do not add up to 100 in each category.

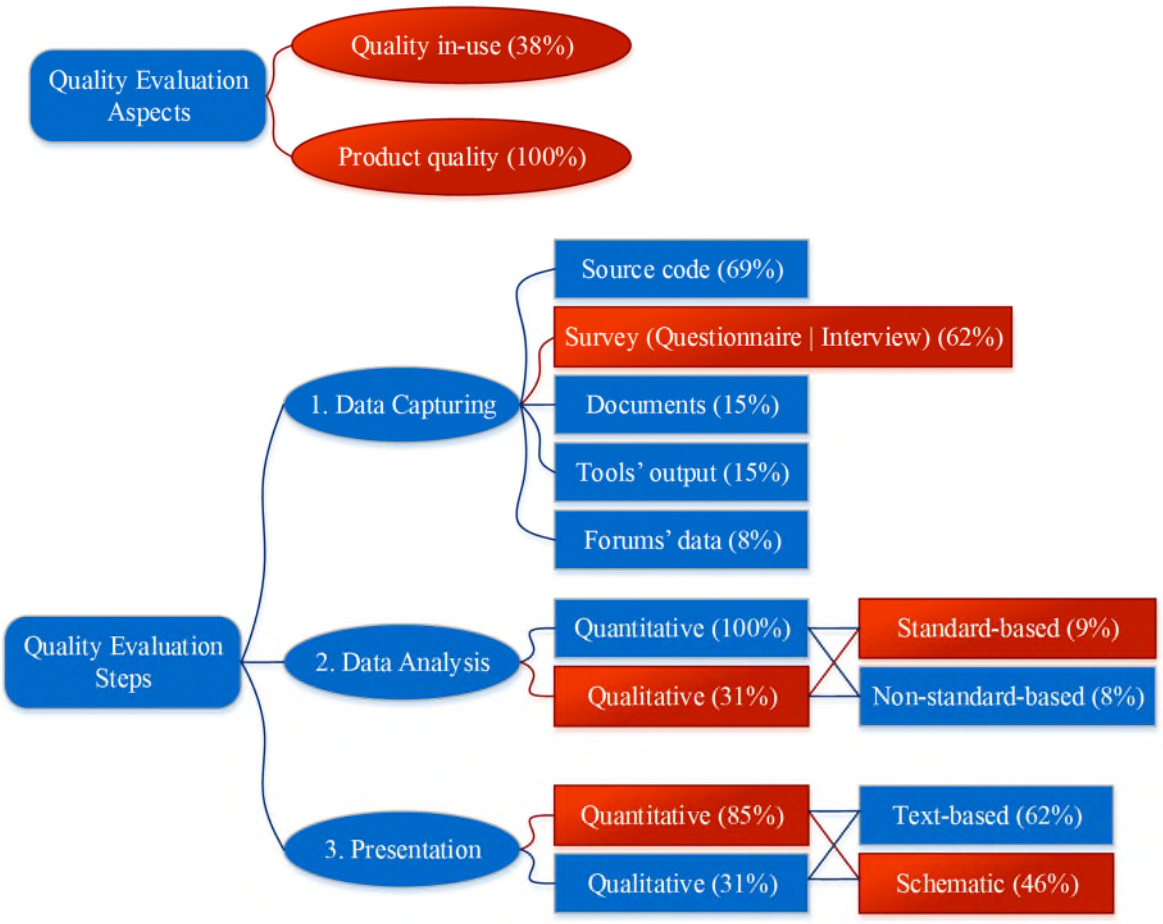


Figure 6: The position of the thesis on the scope: the orange color pieces

1.3 Models, norms, and standards in software quality domain

Many publications in the literature demonstrate the various aspects of software quality assessment. Since 1970's, several software quality models have been proposed such as McCall, Boehm, Dromey, FURPS, ISO/IEC 9126 and ISO/IEC 25000 (SQuaRE).

For easing the professionals to move between projects and products within the organization and reduce the effort required for training, software standards have been presented. A unified method for performing the software project tasks is only possible when all the stakeholders know and understand the standard way of developing and maintaining the software products.

In an industrial context, the standards increase the professionalism by providing access to good practices as defined by the experienced practitioners in the software industry. For example, many

companies benchmark the ISO and IEEE standards as a basis for improving their processes and practices.

The “International Organization for Standardization (ISO) developed the 9000 family of standards to define good practice in the area of quality management systems. Within the ISO 9000 family, the ISO 9001 standard defines the specific set of quality management system requirements. ISO 9001 is the standard that organizations can be certified against [2]. The Software and Systems Engineering Standards Committee of the IEEE Computer Society develops and maintains a set of software engineering guidelines for organizations helping them to define their software processes.

These standards can provide guidance that minimizes time and effort. The ISO 9001 standard and the SEI’s CMMI for Development provide road maps for what should occur in a good software engineering environment while the IEEE Software Engineering Standards provide more detailed “how-to” information and guidance.

A well-managed organization supported by technology is emphasized by SEI. The CMMI is a reference model that covers the evolution of software engineering from an ad hoc context to a disciplined high-performance organization. According to the SEI [23], principal areas of work include ‘management practices’ which focuses on the ability of organizations to predict and control quality, schedule, cost, cycle time, and productivity when acquiring, building, or enhancing software systems. Technical practices focus on the ability of software engineers to analyze, predict, and control selected properties of software systems. The staged representation SEI CMMI for Development is a five-level model that includes a four-level structure of best practices designed to be a roadmap to improving software quality and project performance [2].

The ISO/IEC 25000 Software Engineering—Software Product Quality Requirements and Evaluation (SQuaRE) standard series (transition from the previous ISO/IEC 9126 and 14598 series of standards) provides a reference model and definitions for external and internal quality attributes and quality-in-use attributes. This standards series also provides guidance for specifying requirements, planning, and managing, measuring, and evaluating quality attributes. It was shown that the international standard ISO/IEC 25000 [24] is capable of assessing the quality of a broad range of software applications; from traditional to new application classes such as smart mobile devices [25]. The quality model in ISO/IEC 25010 is designed to identify relevant quality characteristics for software products, which can be used to establish requirements, criteria for

satisfaction and the corresponding measures [26]. ISO/IEC 25010 describes the quality model encompassing the characteristics and sub-characteristics for software quality in use, and software product quality. In addition to the ISO/IEC 25010 that provides the quality characteristics, an approach is also needed to specify the quality measurements. In this work, we applied the ISO/IEC 25021 [27] which provides a set of rules to design and verify the appropriate quality measure elements.

1.4 Data collection for quality evaluation

The objective of the quality evaluation is to determine how well the final product meets the original requirements. Data analysis is pointless without good data. If the right data items are not collected, then the objectives of the measurement program cannot be accomplished [2].

Who should collect the data? Deciding who should collect the data is an important part of ensuring that good data is collected. In most cases, the best choice is the "owner" of the data. The data owner is the person with direct access to the source of the data and in many cases, is actually responsible for generating the data.

How should the data be collected? The best method is 'automation' [2]. The automatic data collecting methods use standardized forms at minimum expense since and maximum validity. In some cases, the data should be collected by asking questions from the stakeholders. In these cases also on-line questionnaire – which is a kind of automation – can be applied [28]. In the next sections, we explain more about the data collection methods and especially on-line questionnaires.

Questionnaires and interviews are the most popular instrument for data collection. [29]. By definition, 'questionnaire' is a set of written questions formed for addressing a specific research question. There are many factors influencing the results of the questionnaires, such as questions wording, questionnaire formatting, the respondents, the generalizability of the results, etc. Pfleeger and Kitchenham presented a six-part series on principles of survey research [30]–[35]. They explained – in detail – how to design, construct, and conduct a survey in addition to the threats and pitfalls in survey administration and data analysis.

Questionnaire-based surveys are becoming popular in software engineering community. According to our simple search in Inspec digital library, in 2007 only 37% of the studies in software engineering have addressed a questionnaire-based survey, while this increased to 63% in 2016, i.e.

70% growth. One of the reasons for this increase can be the features that electronic-based survey tools provide for the researchers. Nowadays, researchers create their electronic and web-based questionnaires efficiently with friendly graphical user interfaces. This makes the creation, global submission, data collection, and results analysis simpler than before when the paper-based questionnaires were being used.

1.5 Quality Profile: Expected vs. Observed quality

Software quality is defined as the comprehensive set of characteristics that enable the product to satisfy the stakeholders' needs. Accordingly, "software quality is fundamental to software success" [36]. Furthermore, evaluation of software quality is essential, since inadequate quality in a software product may lead to human or financial losses [37] while high-quality are fundamental to providing value, and avoiding potential negative effects for the stakeholders.

Taking a broader perspective, high-quality software is recognized as a product that has been specified correctly, and that meets its expected specifications. Software products meeting the stakeholder's requirements are more likely to be accepted and utilized by the stakeholders [2]. Comprehensive specification and evaluation of the quality of software is an essential factor in ensuring value to stakeholders. This added value can be achieved by defining the desired quality characteristics associated with the stakeholders' goals and objectives. These features help to represent the quality of the software products from the perspective of that particular characteristic.

The software quality is evaluated by a collection of relevant quality characteristics which are measured by applying a measurement method. A measurement method is a logical classification of operations applied to quantify attributes with respect to a specified scale. During this process, software quality measures turn into quantifications of the quality characteristics. Not every quality characteristics are of equal importance to a software product. A method will be used to identify the most important quality characteristics by means of a risk assessment, to establish achievement criteria, and to finally measure the quality characteristics using the ISO/IEC 25010 standard. The quality characteristics can be addressed at the beginning of a development process to discover the expected software quality, and at the end of the development process, that leads to software product quality. This can be achieved by interviewing stakeholders inside the project (such as the developers, product manager, the project manager, configuration manager, etc.) and outside the project (the various types of users are important) [38].

In the current research project, the objective is to 1) build the Quality Plan Profile by eliciting the expected quality characteristic based on customer quality requirement engineering. 2) Build the Product Quality Profile by quantifying the quality characteristics of the software products. Through this approach, a number of indicator values will be collected measuring the strength of each quality factor, and 3) create the quality deviation artifact (QDA). The QDA is an artifact that shows the deviation between the planned and the observed software quality. Although the QDA was one of our objectives and we theorized the creation of that, unfortunately we could not have a chance to practically create and test it in form of a case study. It remains as a future work for other researcher to follow and improve our methodology in order to create and analyze the QDA.

The iterative and incremental stepwise process illustrated in Figure 7 represents the steps and the roles in our quality engineering process. The process starts with eliciting the expected customer's quality by the collaboration of quality engineering team and the customer. The next step is to quantify the elicited quality factors that is performed by applying the ISO/IEC 25000 approach. The ISO/IEC 25000 relates the preferences and requirements of stakeholders regarding the software product to the standard software quality characteristics. The stakeholders' needs and requirements are investigated and identified using a structured questionnaire. The questionnaire consists of a number of questions regarding product and process quality characteristics. These characteristics are intended to be expressed in the stakeholders' language. Instead of asking whether "usability" is important, one asks questions about the qualifications of the users that impact the usability requirements, for example, the quantity of users, their involvement with the product (or a similar one), and their educational level. Similar questions are asked about the other quality characteristics. The answers given are used to identify the most relevant quality characteristics and the related values. The detailed steps of the quality measurement process are explained in section III.

The aim of step 3 is to build the quality plan profile. The quality engineering team provides the quality plan profile that is a diagram showing the quantified values of elicited quality requirements.

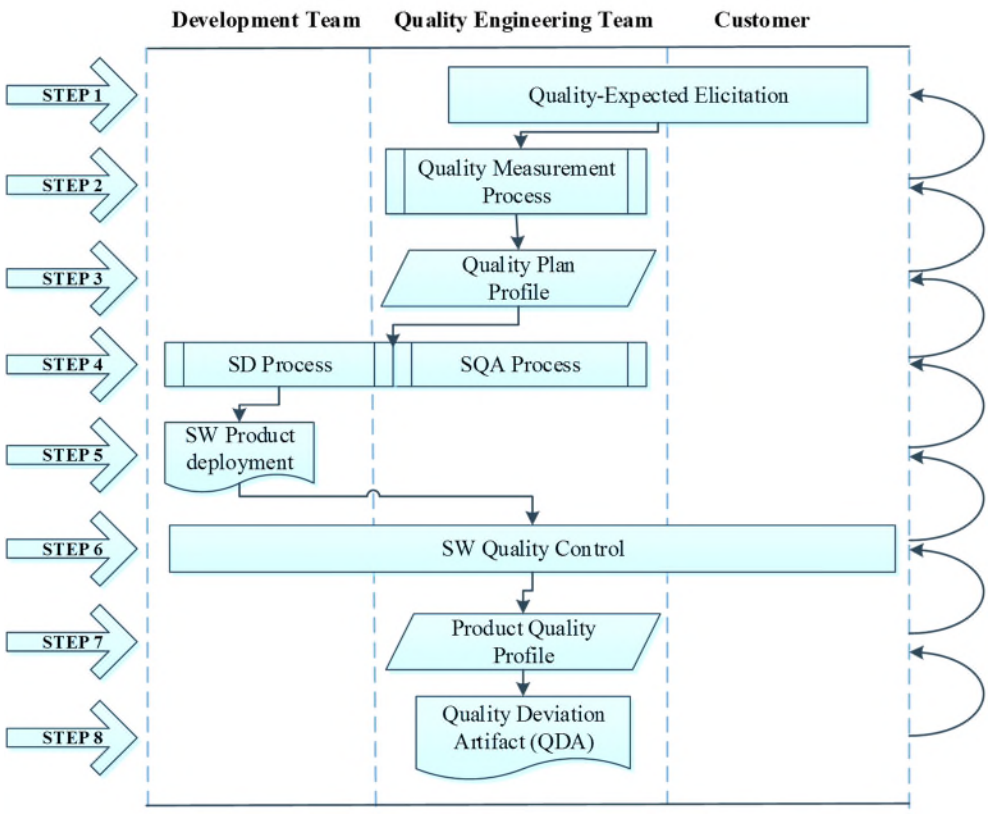


Figure 7: Method to build quality deviation artifact; iterative and incermental

In step 4 in Figure 7 the actual development process takes place. The software developers form the team; the development method is selected, and project management considerations are taken into account. Concurrently the software quality assurance activities are performed by the quality engineering team.

The quality assurance includes the preventive activities that help the development process to come up with a product as flawless as possible. The result of development and quality assurance process is the executable software product that can be deployed on the customer's site, which is shown as step 5 in Figure 7.

After the software product is delivered to the customer, the next step is to measure the product quality by performing the quality control actions, which is step 8 in Figure 7. The goal is to evaluate the expected quality elements that had been highlighted during the quality planning phases; i.e. step 1 through step 3 in Figure 7.

1.6 Research Approach

The research of this thesis is conducted in a survey-based style using an empirical strategy with an exploratory approach. There are two types of research paradigms that have different approaches to empirical studies: Explanatory and Exploratory. Exploratory research is concerned with studying objects in their natural setting and letting the findings emerge from the observations. We also limited our project borders to empirical research since we conducted our case studies in industrial context using the real software users. “There are three major different types of strategies in empirical contexts: survey, case study, and experiment. A survey is a systemic method for collecting data and information from or about people to describe, compare or explain their knowledge, attitudes, and behavior. The primary means of gathering qualitative or quantitative data are interviews or questionnaires. These are done through taking a sample which is representative of the population to be studied. The results from the survey are then analyzed to conclusions. They are then generalized to the population from which the sample was taken [39]”

1.7 Objective of the Research

In this thesis, we aim to address the quality of the software product from the users’ point of view. The goal is to create the expected and observed quality profile of the software product. This can help to demonstrate the differences between what qualities were expected from the users’ side, and what qualities are observed in the software product. We employed an empirical strategy using a survey-based method to create the profile of expected and observed quality characteristics. After developing a standard-based quality model, we created four types of questionnaires for end users and power users, which aim to elicit the quality factors. As a case study, we conducted three surveys in two phases in the industry. In phase I, we asked the potential users of a software product which was under development, to answer the ‘expected-quality’ questionnaire. In phase II, the users of an existing software product were asked to answer the ‘observed-quality’ questionnaire.

1.8 Structure of the thesis

The following chapter presents the literature review that systematically shows how previous publications explain two subjects: the difficulties in the conducting the surveys in software engineering field, and the effects of the participants’ knowledge on the results of the surveys. Chapter 3 presents the methodological approaches used in our research. Chapter 4 presents a

synthesis of the publications made in the framework of this research project and how these publications fit together to form our synthesis. Chapter 5 presents a discussion of the publications. Chapter 6 summarizes the findings of this research project and provides recommendations for future research in the field. Finally, the complete texts of the published articles are included in the appendices.

CHAPTER 2 LITERATURE REVIEW

To better understand how the practitioners are evaluating the software quality from the user's point of view, we planned to review the related literature retrieving the challenges in the field. For this purpose, we conducted the literature review for two subjects. First, we tried to retrieve the challenges in evaluating the software quality using the surveys, since using the surveys is the most popular instrument of data capturing for quality evaluation projects. Analyzing the results of the first literature review led us to perform the second review focused on one of the found challenges: “participants’ knowledge”.

2.1 The difficulties in surveys for software quality evaluation

2.1.1 Goal

To know how the researchers in software engineering domain conduct the surveys and to know the difficulties that they faced with, and how they cope with the issues.

2.1.2 Data extraction: The targeted context values

Our literature review aims to extract the following context values from the selected papers:

- Has any statistical test been applied to the hypothesis or to the survey results?
- What is the research question (RQ) or the message of the paper?
- What validity analysis has been done in the report?
- Has the paper used the Likert scales for quality evaluation?
- What is the number of the people who participated in the survey?
- How many questions are there in the questionnaire?
- Was the questionnaire pre-tested before being published?
- Were the participants cherry-picked for the survey? How?
- Was the "Knowledge of the participants" addressed in the survey process?

And other related data that may help us during our analysis. For each paper, we determined the data lacks. The lacks are the context values that have not reported in the paper.

2.1.3 Strategy

A literature review (LR) is performed to retrieve the related reports. We performed the LR in two styles: a *systematic* literature review and a *wild* literature review. In the systematic style, we define

the sources, created a comprehensive search string and then we retrieved and refined the data. In the wild style, we did a search, generally on Google scholar, for the specific author(s) or specific paper(s) that we guessed have some related information.

In the next sections, the process, the results and the synthesis are presented.

2.1.4 Search string

We defined a search string base on the goal of the LR. Our search string is formulated as:

("Software engineering" WN KY)
AND
(("survey" WN TI) OR ("questionnaire") WN TI)
AND
(("problem" WN KY) OR ("difficulty" WN KY) OR ("difficulties" WN KY) OR ("issue" WN KY) OR ("issues" WN KY))
AND
({ca} OR {ja} WN DT)
AND
(2007 OR 2008 OR 2009 OR 2010 OR 2011 OR 2012 OR 2013 OR 2014 OR 2015 OR 2016 OR 2017 WN YR)
AND
({english} WN LA)

As it can be seen we limited our search to English conference or journal papers that were published since the year of 2007.

2.1.5 Libraries

The search string was applied to the most famous digital libraries. The libraries are:

- *Compendex* (14 million references),
- *Inspec* (16 million references),
- *ACM* (the most comprehensive collection of full-text articles),
- *IEEE-Xplore* (nearly 2 million full-text documents and 12K+ new papers every month),
- *Web of Science* (90 million records).

2.1.6 The retrieved papers

Applying the search string to the libraries resulted in retrieving 216 records. In the first step, we excluded the papers with the following conditions:

- duplicate papers
- less than 5 pages

- titles not related
- abstract not related
- full text not related
- PDF unavailable

The results are presented in Table 8. Finally, 27 related papers were retrieved.

Table 8: Number of retrieved papers for LR

	Compendex/ Inspec	**ACM**	**IEEE Xplore**	**WEB of Science**	**Wild (Google Scholar)**	**Total**
Total retriever	136	5	26	36	13	**216**
Duplicates	27	1	4	1	0	33
Pages<5	10	2	9	4	5	30
Title not-related	33	0	0	0	0	33
Abstract or body not-related	56	0	11	17	3	87
PDF unavailable	0	0	1	5	0	6
Related	10	2	1	9	5	27

The 27 related papers are listed in the following table (Table 9):

Table 9 : List of 27 retrieved papers for the 1st phase of the LR

id	Title of the paper	firstAuthor_year
Descriptions: Paper's context values		
L_1	A state-of-the-practice survey of risk management in development with off-the-shelf software components [40]	Li_2008
This paper presents the impacts of Off-the-shelf components on software quality. The questionnaire in this project, that includes 20 questions, uses the five-point Likert scales to collect background information about the participant company and the respondents. Questions are especially on risk management and process improvement. The researchers used stratified random sampling to select companies and convenience sampling to select projects inside a company. 133 projects from 3 countries were selected for the survey. The researchers contacted 1,087 companies in three countries. 39% were software companies and had experience with OTS component-based development. 55% companies declared a willingness to participate in the study and the remaining claimed that they were too busy to participate. They sent the questionnaire to these 234 companies. 54% of the companies (133) responded. The questionnaire was pre-tested using a paper version by 10 internal experts and eight industrial respondents before being published. The research process and the results are validated well. The hierarchical multiple regressions have been done on the results. **Lack:** It seems that the knowledge of the participants is not addressed in this survey.		
L_2	A survey into the rigor of UML use and its perceived impact on quality and productivity [41]	Nugroho_2008
In this paper, the styles and rigor in UML modeling (i.e., completeness, the level of detail, and correspondence with implementation) and how software engineers perceive these, as well as the perceived impacts of using UML on productivity in various phases of software development are discussed. The 20-question questionnaire is aimed to collect data and investigate the opinions from professional software engineers about different styles used in modeling and their perceived impacts on quality and productivity in software development. 80 researchers participated in this survey. The research focused on a specific group of participants because the authors claim that "the main target of the survey was the software developer because they believe that they are the ones who are responsible for implementing UML models". The research process and the results are internally and externally validated. **Lack:** It seems that the knowledge of the participants is not addressed in this survey.		
L_3	A Survey of Metrics Use in Finnish Software Companies [42]	Soini_2011

Table 9 : List of 27 retrieved papers for the 1st phase of the LR

id	Title of the paper	firstAuthor_year
Descriptions: Paper's context values		
"This paper uses the results of an empirical case study to examine how the measurement was implemented in practice from the perspective of the software process". A questionnaire with 13 questions was submitted to 40 Finnish software companies. The questionnaire aimed to collect separate metrics of customer feedback information. **Lack:** statistical tests, validity checks, pilot conduction, and neglecting on how to select the participants and their related knowledge are the items that are lacking in this paper.		
L_4	A survey of software development with open source components in Chinese software industry [43]	Chen_2007
This paper addressed the issues in reusing open-source components. The questionnaire aims to collect information "on three main issues in reusing OSS components for software development in Chinese software industry, namely component selection, licensing terms, and system maintenance." The survey conducted on 47 projects using a 35-questions questionnaire. The researchers conducted a pre-study to verify and refine the preliminary questionnaire (Individual interview + A group discussion). The survey is well validated in terms of Construct, Internal, and External validity. **Lack:** statistical tests, and neglecting on how to select the participants and their related knowledge are the items that are lacking in this paper.		
L_5	A survey on the business relationship between Chinese outsourcing software suppliers and their outsourcers [44]	Li_2007
This paper reviews the relationship between the Chinese suppliers and outsourcers. The questionnaire aims to collect data and "investigate how Chinese suppliers have built and maintained partnership or contract relationship with their outsourcers, and the effect of these relationships on the success of outsourcing". The questionnaire includes 66 questions that were distributed to 53 projects. The results are statistically tested by ANOVA and the validity is checked in terms of Construct, Internal, and External validity. **Lack:** neglecting on how to select the participants and their related knowledge are the items that are lacking in this paper.		
L_6	Adoption of OSS development practices by the software industry: A survey [45]	Petrinja_2012
This paper aims to "collect data related to practices and elements in the development process of companies that influence the trust in the quality of the product by potential adopters". The questionnaire consists of 53 questions that distributed to 56 projects managers. **Lack:** statistical tests, validity checks, pilot conduction, and neglecting on how to select the participants and their related knowledge are the items that are lacking in this paper.		

Table 9 : List of 27 retrieved papers for the 1st phase of the LR

id	Title of the paper	firstAuthor_year
Descriptions: Paper's context values		
L_7	Applying mLearning in software engineering education: a survey of mobile usage [46]	Macphail_2012
This paper aims to show how mLearning is suitable for teaching. It uses a questionnaire that distributed to 321 participants. The results were tested by Mann-Whitney U test. **Lack:** validity checks, pilot conduction, and neglecting on how to select the participants and their related knowledge are the items that are lacking in this paper.		
L_8	Effort estimation in Agile software development: A survey on the state of the practice [47]	Usman_2015
This paper aims to "report on the state of the practice on effort estimation in agile software development". The questionnaire aims to collect data on effort estimation techniques from agile teams. The results were validated in terms of Construct, Internal, External, and Conclusion validity. 63 responses were received from the participants. The questionnaire was piloted. It was "iteratively designed and updated by the authors". **Lack:** statistical tests, and neglecting on how to select the participants and their related knowledge are the items that are lacking in this paper.		
L_9	Identifying research gaps in requirements engineering education: An analysis of a conceptual model and survey results [48]	Memon_2012
In this paper, a questionnaire was created to investigate lecturers' perceptions of the Requirement Engineering Education problems presented in integrated viewpoint. The survey was conducted on 18 participants. "The data collection was aimed at lecturers who have taught RE course. The respondents had between 5-20 years teaching experience." This study seems to take the knowledge of the participants into consideration. **Lack:** statistical tests, validity checks, pilot conduction are the items that are lacking in this paper.		
L_{10}	Towards understanding software change request assignment: A survey with practitioners [49]	Cavalcanti_2013

Table 9 : List of 27 retrieved papers for the 1st phase of the LR

id	Title of the paper	firstAuthor_year
Descriptions: Paper's context values		
This paper reviews the impact of Change Request assignment on software development. Using 38 questions, 36 participants participated in the survey. A pilot survey was done since the researchers claim that "We constructed an initial version of the questionnaire and the resulting instrument was validated in three rounds: two rounds for critiques, and one round for a pilot survey". Participants cherry picked and their knowledge has been taken into consideration because it is claimed that "As a condition to participate in the survey, the software developers should be involved with Change Request assignment in their daily activities". **Lack:** statistical tests and validity checks are the items that are lacking in this paper.		
L_{11}	Requirements reuse and requirement patterns: a state of the practice survey [50]	Palomares_2016
This paper is "an exploratory study of the practices in requirements reuse that are currently being used in organizations and to study in more depth the possible benefits and drawbacks of the use of patterns as a requirement reuse technique". 33 questions are created to collect data about requirements reuse and requirement patterns. 71 participants participated in the survey. Chi-Square exact test is used to validate the results. In addition, internal, external, construct validations have been performed. Using sampling validity, the participants were picked for the survey. **Lack:** the related knowledge of the participants is the item that is lacking in this paper.		
L_{12}	A survey on quality attributes in service-based systems [51]	Ameller_2016
This paper aims to review the quality attributes in service-based systems. The questionnaire collects the data about the significance of quality attributes when designing service-based systems and how quality attributes are addressed through design decisions. The Fisher Test is done to validate the results. The research process and results are also validated in terms of external, randomization, and exclusion validity. The questionnaire uses the Likert scales and has 30 questions. 56 responses have received for this survey. Selecting the participants have been done with care; "we included researchers with practical design experience". The research was pre-tested since it is claimed that "we piloted the data collection instrument in multiple iterations until potential respondents understood our questions and intentions". The knowledge of the participants is taken into consideration partially since it is claimed: "Our population was the global community of software engineering practitioners, as well as researchers that have practical experience with, and knowledge about, designing Service-based systems". Lack: it seems that there no lack.		
L_{13}	Questionnaire-based risk assessment scheme for Japanese offshore software outsourcing [52]	Tsuji_2007

Table 9 : List of 27 retrieved papers for the 1st phase of the LR

id	Title of the paper	firstAuthor_year
Descriptions: Paper's context values		
The research question is formulated as "Although a lot of engineers have experienced the success and failure of their projects, their know-how still remains as tacit knowledge. This paper proposes a risk assessment scheme for new projects by externalizing such tacit knowledge". The paper uses the statistical tests as frequency analysis, structural equation modeling, and conjoint analysis. 175 responses were received. The survey was pre-analyzed: "Based on the pre-analysis, we choose three property types for describing software development instead of the previous nine attributes". **Lack:** validity checks, and neglecting on how to select the participants and their related knowledge are the items that are lacking in this paper.		
L_{14}	Freemium business model: construct development and measurement validation [53]	Hao-Chen_2015
"The purpose of this paper is to probe into the development of the dimensions of the freemium business model and validate the measurement". The t-test is performed to validate the process and the results. In this research, 1061 responses were received. The study is pre-tested: "This study provided 30 pretest items and invited experts, scholars and business workers to screen or review the questionnaire content, dimensions, and terms. The experts and scholars were asked to evaluate each item as being appropriate, appropriate after revision, or inappropriate, and then provide their opinions for revision". The participants were picked by care. They first interviewed and then the questionnaire was submitted. Convenience sampling is also performed. **Lack:** It can be said that the researchers were aware of the importance of the knowledge of participants, but they didn't get into that specifically.		
L_{15}	Software Engineering Researchers' Attitudes on Case Studies and Experiments: an Exploratory Survey [54]	Tofan_2011
This paper aims to answer that "How do empirical software engineering researchers perceive the differences between case studies and experiments? And how do perceptions of researchers vary along their views on the nature of case study". The researchers have used Mann--Whitney U--test, as well as the External, Internal, and Construct validity checks. 26 SE researchers participated in answering to 28 questions of the questionnaire. The research pre-tested the survey in three iterations. **Lack:** neglecting on how to select the participants and their related knowledge are the items that are lacking in this paper.		
L_{16}	Instructor's Acceptance of Games Utilization in Undergraduate Software Engineering Education A Pilot Study in Turkey [55]	Albayrak_2015

Table 9 : List of 27 retrieved papers for the 1st phase of the LR

id	Title of the paper	firstAuthor_year
Descriptions: Paper's context values		
The paper revealed that 'the number of hours per week the instructor plays game', 'instructor's experience in using games for educational purposes in general', and 'instructor's experience in designing games' have a significant impact on the instructor's decision to use games in software engineering education". The researchers performed the t-test. 30 persons participated in the survey to answer the 10 questions. Likert-type questions were used. **Lack:** validity checks, pilot conduction, and neglecting on how to select the participants and their related knowledge are the items that are lacking in this paper.		
L_{17}	Presenting software engineering results using structured abstracts: a randomized experiment [56]	Budgen_2008
The paper shows "whether structured abstracts are more complete and easier to understand than non-structured abstracts for papers that describe software engineering experiments. […] The 64 participants were each presented with one abstract in its original unstructured form and one in a structured form, and for each one were asked to assess its clarity (measured on a scale of 1 to 10) and completeness (measured with a questionnaire that used 18 items). The Construct Validity and Internal Validity were also done. **Lack:** statistical test, pilot conduction, and neglecting on how to select the participants and their related knowledge are the items that are lacking in this paper.		
L_{18}	Towards understanding the underlying structure of motivational factors for software engineers to guide the definition of motivational programs [57]	da Silva_2010
This paper attempts to present a better understanding of motivation in software engineering. The process is validated in terms of external and construct validity.176 participants responded to 3 questions in the questionnaire. **Lack:** statistical tests, pilot conduction, and neglecting on how to select the participants and their related knowledge are the items that are lacking in this paper.		
L_{19}	A look at typical difficulties in practical software development from the developer perspective a field study and a first solution proposal with UPEX [58]	Erfurth_2007
This paper just "presents a look at typical difficulties in practical software development from the developer perspective. 65 participants participated in this survey". "Goal of the questionnaire was to get answers regarding the state of art in practice." The researchers received 65 answered questionnaires from 54 companies. **Lack:** statistical tests, pilot conduction, and neglecting on how to select the participants and their related knowledge are the items that are lacking in this paper.		

Table 9 : List of 27 retrieved papers for the 1st phase of the LR

id	Title of the paper	firstAuthor_year
Descriptions: Paper's context values		
L_{20}	On the effectiveness of early life cycle defect prediction with Bayesian Nets [59]	Fenton_2007
This paper discusses an experiment to develop a causal model (Bayesian net) for predicting the number of residual defects that are likely to be found during independent testing or operational usage. A questionnaire for 31 projects distributed to the software project managers to validate the model. The questionnaire has 10 questions and it is Externally validated by the authors. **Lack:** statistical tests, pilot conduction, and neglecting on how to select the participants and their related knowledge are the items that are lacking in this paper.		
L_{21}	Use of questionnaire-based appraisal to improve the software acquisition process in small and medium enterprises [60]	Garcia_2008
This paper aims to show the application of a "Maturity Questionnaire" in a disciplined way. The proposed questionnaire focuses in Supplier Agreement Management Process Area of the CMMI. The questionnaire distributed to 600 participants. **Lack:** statistical tests, pilot conduction, validation checks, and neglecting on how to select the participants and their related knowledge are the items that are lacking in this paper.		
L_{22}	Empirical Assessment of MDE in Industry [61]	Hutchinson_2011
This paper aims to review the defects that are likely to be found during independent testing or operational usage. Using largely qualitative questionnaire the researchers investigate a range of technical, organizational and social factors that influence organizational responses to Model Driven Engineering. Lack: No of questions, population size, statistical tests, validity checks, pilot conduction, and neglecting on how to select the participants and their related knowledge are the items that are lacking in this paper.		
L_{23}	Personality, emotional intelligence and work preferences in software engineering: An empirical study [62]	Kosti_2014
This paper tries to show that "The associations can help managers to predict and adapt projects and tasks to available staff. The results also show that the Emotional Intelligence instrument can be predictive". In this survey, 20 questions distributed to 272 participants. ANOVA test and Conclusion validity is performed on the results. **Lack:** pilot conduction, and neglecting on how to select the participants and their related knowledge are the items that are lacking in this paper.		

Table 9 : List of 27 retrieved papers for the 1st phase of the LR

id	Title of the paper	firstAuthor_year
Descriptions: Paper's context values		
L_{24}	Code Readability Testing, an Empirical Study [63]	Sedano_2016
This paper tries to show that Code Readability Testing improves programmers' ability to write readable codes. A questionnaire is created to retrieve the programmers' perspective. The questionnaire distributed to 21 SE master students. Construct, Internal, and External validity is performed on this research. **Lack:** No of questions, statistical tests, pilot conduction, and neglecting on how to select the participants and their related knowledge are the items that are lacking in this paper.		
L_{25}	An Exploratory Study on Open Conversation [64]	Kevin_2011
The goal of the paper is explained as: "In a traditional collocated Software Engineering setting, one of the most important communication patterns is a conversation. Technological support to have conversations in a distributed setting is commonly used, however overhearing conversations of your colleagues is mostly not feasible with these tools. To explore the importance of overhearing conversations we conducted a focus group and a questionnaire in a large international software development company." The 5- point Likert scale questions distributed to 44 participants. The results are tests by Friedman, Wilcoxon test. "Limitation" of research and the results are discussed. **Lack:** No of questions, pilot conduction, and neglecting on how to select the participants and their related knowledge are the items that are lacking in this paper.		
L_{26}	Problems and Opportunities for Model-Centric Versus Code-Centric Software Development: A Survey of Software Professionals [65]	Andrew_2008
"The aim of the survey was to uncover their attitudes and experiences regarding software modeling and development approaches that avoid modeling". 18 Likert-scaled questions distributed to 118 participants. The results were tests by t-test. **Lack:** validity checks, pilot conduction, and neglecting on how to select the participants and their related knowledge are the items that are lacking in this paper.		
L_{27}	Some Lessons Learned in Conducting Software Engineering Surveys in China [66]	Junzhong_2008

Table 9 : List of 27 retrieved papers for the 1st phase of the LR

id	Title of the paper	firstAuthor_year
Descriptions: Paper's context values		
This paper reports on the lessons learned while conducting the surveys. The paper doesn't intend to describe a survey process. Thus, there no context value in it. The author says: "we encountered many difficulties in conducting the surveys, but in most cases managed to find working solutions. We report on the lessons learned while conducting these surveys. We address issues relating to sampling, contacting respondents, data collection, and data validation. The main lessons are: 1) it was necessary to cooperate with a third-party organization with close relations to Chinese software companies; 2) it was necessary to assign researchers to this third-party organization to facilitate data collection and to control the quality of the data collected; and 3) An email survey, after an initial telephone call to establish contact, was the best method for getting questionnaires completed by Chinese respondents.		

2.1.7 The reported difficulties

By analyzing the papers, we retrieved the following list of threats. The reported threats are:

- "Since the participants knew that they were being observed, they may have altered their behavior."
- "The fact that we did not perform a probabilistic sampling for subject selection, has made it difficult to control the distribution of the respondents."
- "The use of self-reporting to collect data on attitude or behavior might have to deal with the accuracy of the information provided by the respondents."
- "Nearly 10% of the questions and alternatives in the final questionnaire were revised based on the pre-study."
- "*Instrumentation Validity*: Instrumentation threats can appear if the experiment has an error in its design or changes are necessary for the middle of the experiment."
- "*Sampling Validity*: The population we were interested in was requirements engineers with industrial experience. Given their professional scope and skills, we may assume that they

were an Internet-aware population with enough expertise to answer an online survey without technological impediments."

- "*Participants' Perceptions*: Since we were using online questionnaires, if the survey proposal was perceived to be junk publicity, the outcome of the results could be affected. To avoid this, our survey invitation e-mails were sent to a conference attendees and the communities from our academic e-mail addresses. We included a brief letter explaining the academic purpose of the survey and provided its link, and signed the e-mails with the authors' names. We also tried to provide a professional image by opening a specific web page."
- "*Low Response Rate*: one of the main problems of online surveys is having very low participation rates."
- "*Low Completion Rate*: the results represent a low completion rate. As a consequence of the survey pilots carried out, we realized that a high percentage of the non-completed attempts occurred in a specific section of the survey owing to a design error in the interface. Before the change was implemented, the completion rate was about 45 %; after the change, it grew to approximately 72 %."
- "*Low completion rate*: [The length and complexity of the survey]. Because of that, we decided not to use quantitative metrics when asking for aspects like requirements reuse, reducing the time needed to answer some questions."
- "*Results Accuracy*: As quantitative measures were not used in the answers for some questions in the survey, the accuracy of the results of such questions might have been affected."
- "*Face Validity*: Face validity is the extent to which a measure addresses the desired concept, i.e. whether it measures what it is supposed to measure. In order to ensure face validity, we discussed with our collaborators whether the proposed survey questions were a good representation to answer the research questions (RQs)."
- "*Exclusion:* means that participants who are not sufficiently experienced were excluded from the study."

- "*Randomization* means that we used a sampling technique which leads to random participants."
- "The Survey's external validity is affected by the small size of our sample that included only 26 subjects. Moreover, all participants attended the summer school on empirical research methods, thus potentially influencing their perceptions. The sessions at the summer school covered experiments extensively, while case studies received little attention."

2.1.8 Conclusion

Protocol

After analyzing the related retrieved papers, the results show that almost all the reports have followed similar steps for conducting the surveys. We should emphasize that the steps presented by Kitchenham and Pfleeger (2013) have been used as a reference for many surveys in SE domain. The steps, namely Survey Protocol are listed as the following 6 steps:

1. Creating the questions that cover the research questions
2. Selecting the appropriate multiple-choice options for each question
3. Conduct a pilot survey (In some cases) to refine the questions
4. Selecting the survey channel, i.e. the tool for implementing the questionnaire
5. Selecting the appropriate population and sampling
6. Analysis the responses

Validity

Almost half of the retrieved papers presented a validity analysis on the process and the results of the surveys. They presented a combination of *Construct*, *Internal*, *External*, and *Conclusion* validity. On the other hand, some papers have performed a statistical test on the survey results. The tests include ANOVA, chi-square, Fisher, and Man-Whitney tests.

Our LR shows that the papers that presented a validity check, they have also applied a statistical test to the results of the surveys.

Using the Likert scales

The results of our LR shows that few papers (11 papers out of 27) have used Likert scales in their questionnaires. The reason may be the difficulties in analyzing the data that the researchers will have while using the ordinal scales.

Population size

Regardless of three exceptions, it is shown that in average, about 80 responses have been received for each questionnaire. Normally the researchers ask a bigger set of participants, but finally, around 80 valid responses have been received in average. There is three exceptions; 1) L_{14}: Hao-chen_2016 that received 1061 responses, 2) L_{21}: Garcia_2008 that received 600 responses, and 3) L_{23}: Kosti_2014 that received 272 responses.

Questionnaire size

Among the retrieved papers, there are 25 to 30 questions in average. Two questionnaires have 66 and 63 questions and one questionnaire has only 3 questions.

Pilot Survey

Half of the retrieved papers reported that they have conducted a pilot survey (pre-test) before the questionnaires are published. The researchers believe that refinement of the questions before final conduction to the public group will help to eliminate the irrelevant questions or disambiguation.

Picking the participants

Attention to '*who should participate*' and '*who should not'*, is neglected in the literature. Only a few researchers have emphasized on the related knowledge of the participants. In the best case, the researchers tried to select the most appropriate focus group, e.g. developers, instructors, students, etc. But picking the individual participants is often neglected.

2.2 The participants' knowledge

2.2.1 Goal

To investigate the impacts of "participants' knowledge" on the surveys in software engineering.

2.2.2 Data extraction: The targeted context values

From the retrieved papers the following data were extracted:

- Research question and/or the message of the paper
- The part of the papers that the authors discussed the role of participants' knowledge in their surveys

2.2.3 Strategy

A literature review (LR) is performed to retrieve the related reports. In this stage, a wild literature review has been performed.

2.2.4 Search string

A simple search string was used to retrieve the related papers:

"Software Engineering" AND
"Questionnaire" AND
"Participant's' knowledge"

We limited our search to English conference or journal papers that were published since the year of 2007.

2.2.5 Libraries

The retrieved papers were extracted from the libraries of the publishers:

- IEEE
- ELSEVIER
- ACM
- SPRINGER
- SciKA

2.2.6 The retrieved papers

Applying the search string to the libraries resulted in retrieving 240 records. In the first step, we excluded the papers with the following conditions:

- duplicate papers
- less than 5 pages
- titles not related
- abstract not related
- full text not related
- PDF unavailable

Finally, 11 related papers were retrieved. The results are presented in the table below (Table 10).

Table 10: List of 11 retrieved papers for 2nd phase of the LR

id	Title of the paper	firstAuthor_year	Type
Descriptions: Paper's context values			
K_1	Automated Unit Test Generation during Software Development: A Controlled Experiment and Think-Aloud Observations [67]	Rojas_2015	Type I
The authors discussed the knowledge of the participants: "Participants without sufficient knowledge of Java and JUnit may affect the results; therefore, we only accepted participants with past experience, and we provided the tutorial before the experiment. In the background questionnaire, we included five JUnit and five Java quiz questions. On average, 6.93 out of these 10 questions were answered correctly, which strengthens our belief that the existing knowledge was sufficient." However, the authors did not investigate the effect of participants' knowledge on the results of their research project.			
K_2	Comparing attack trees and misuse cases in an industrial setting [68]	Karpati_2014	Type II

Table 10: List of 11 retrieved papers for 2nd phase of the LR

id	Title of the paper	firstAuthor_year	Type
Descriptions: Paper's context values			
The participants were randomly assigned to one of four experiment groups. The participants in groups 1 and 2 used Misuse Cases before Attack Trees, whereas groups 3 and 4 used them in the opposite order. To control for differences between the four experiment-groups, the researchers collected data about each participant's background in the pre-task questionnaire. A set of Control variables is defined in this research including the Knowledge and the Job. Each participant is asked to report his/her knowledge. The participants' backgrounds can explain why the participants perceived Attack Tree as more useful, yet had stronger intentions to use Misuse Cases in the future. The authors of the paper have not explained about the effect of Knowledge on the survey results.			
K_3	Identification of aspect candidates by inspecting use cases descriptions [69]	Campos_2010	Type III
The first case study was carried out with 19 voluntary students of the Software Engineering II discipline. The second study was carried out with 7 voluntary students that were taking the Software Engineering discipline, as part of their Master's Degree program in Computer Science. The participants of both case studies had basic knowledge about UML use case technique. However, the second group had a more heterogeneous education, since they are students with different backgrounds, which means that the knowledge about the UML use case technique had been achieved from different books and techniques. The first group, on the other hand, had achieved basic knowledge about use cases in the same course and in the same college term. Results: The participants that had some knowledge about AOP (aspect-oriented programming) tried to identify the aspect candidates intuitively, using their present knowledge. Results: By the analysis it is also possible to say that, after the training, the knowledge about AOP did not affect the number of aspects identified in each group.			

Table 10: List of 11 retrieved papers for 2nd phase of the LR

id	Title of the paper	firstAuthor_year	Type
Descriptions: Paper's context values			
K_4	Design Patterns in Software Maintenance: An Experiment Replication at Freie Universitat Berlin [70]	Prechelt_2011	Type II
The experiment was conducted then a short course on design patterns to increase the participants' knowledge of design patterns. The researchers asked for prior knowledge of 17 particular design patterns on a scale from 1, 2, or 3 (described as "never heard of it", "have only heard of it", and "understand it roughly", respectively) up to 7 ("understand it well and have worked with it many times"). Although the researchers notice the importance of the participants' knowledge, no relation between the Participants knowledge and the results of the study is reported.			
K_5	ERP adoption cost factors identification and classification: a study in SMEs [71]	SciKA_2013	Type II
The collected data was based on the participants' knowledge and experience from completed ERP projects in SMEs. The researchers noticed the importance of the participants' knowledge, but they didn't show any relation between knowledge and the results of their studies.			
K_6	Evaluating a Model of Software Managers' Information Needs – An Experiment [72]	Jedlitschka_2010	Type III

Table 10: List of 11 retrieved papers for 2nd phase of the LR

id	Title of the paper	firstAuthor_year	Type
Descriptions: Paper's context values			
Experience: the researchers asked respondents regarding their involvement in the decision-making process, their general business experience in years as well as their experience with technology selection, and their knowledge regarding the technologies. The researcher tested whether role, involvement in the technology selection process, experience, knowledge of the technology, or reading time had an influence on the results. With regard to the participants' knowledge regarding the technology, the researchers asked whether they knew structural testing before and whether they had used structural testing in projects. Only two out of 20 participants had not known structural testing before. Half of the participants had applied structural testing in projects (ten out of 20); these groups were used for the analysis. The analysis did not reveal any significant differences, neither for the whole group nor for the split sample. Result: We can argue that project experience with structural testing does not have an influence on the dependent variables.			
K_7	Evaluation of Software Visualization Tools: Lessons Learned [73]	Sensalire_2009	Type I

Table 10: List of 11 retrieved papers for 2nd phase of the LR

id	Title of the paper	firstAuthor_year	Type
Descriptions: Paper's context values			
Accepting participants with limited knowledge of that IDE would greatly affect the results of the study. Precious time will be spent trying to bring the participants knowledge to an acceptable level instead of carrying out the core experiment. Pre-selection can be done with the aid of a questionnaire that asks about the knowledge of the willing participants [Sensalire and Ogao 2007b; Sensalire and Ogao 2007a]. We found this method better than pre-selection based on the professional level (years of experience in the field). People with identical amounts of experience years can have widely different skills, as it was evident in the software evolution study [Telea 2008].			
K_8	Need of Software Engineering Methods for High Performance Computing Applications [74]	Schmidberger_2012	Type II
In order to reach our goal, the researchers focused on four major topics of the survey, one of them is "background knowledge and education of the surveyed persons". The relation between the participants' knowledge and the results of the study is not reported.			
K_9	On the Relation between Class-Count and Modeling Effort [41]	Nugroho_2008	Type I

Table 10: List of 11 retrieved papers for 2nd phase of the LR

id	Title of the paper	firstAuthor_year	Type
Descriptions: Paper's context values			
Although the participants seem to have moderate industrial experience with UML, their knowledge and current education/ training focus have made them eligible for participating in the experiment. The researchers investigated whether the participant's expertise affects class-count in the models. To this aim, the researchers perform another correlation analysis between the participants' knowledge background and experiences. The researchers use the data obtained from the participant characterization questionnaire in Experiment A. In the questionnaire, the researchers asked a couple of multiple-answer questions related to participants' knowledge and experience. The result in Table 2 shows that there is a positive correlation between the participant's knowledge/experience score and class-count in the models.			
K_{10}	Presenting software engineering results using structured abstracts: a randomised experiment [56]	Budgen_2008	Type III

Table 10: List of 11 retrieved papers for 2nd phase of the LR

id	Title of the paper	firstAuthor_year	Type
Descriptions: Paper's context values			
Based on a regression analysis that adjusted for participant, [...] knowledge of structured abstracts. [...], the use of structured abstracts increased the completeness score and the clarity score. Two dummy variables were constructed to summarize whether the participant had prior knowledge of structured abstracts: Known had a value 1 if the participant knew anything about structured abstracts and 0 if the subject did not know anything or did not answer the question. Not Known had a value 1 if the participant did not know anything about structured abstracts and 0 if the participant knew about structured abstracts or did not answer the question. The researchers performed a more detailed analysis of the results to assess whether any factors such as the knowledge of structured abstracts might have biased the results. Fifty-seven participants answered questions about their knowledge of structured abstracts. Overall (72% said they had known about structured abstracts before taking part in the experiment. None of the uncontrolled factors (preference, role, knowledge of structured abstracts) had a significant effect on the dependent variable. (Total number of years answers to the 18 completeness questions)			
K_{11}	Some Lessons Learned in Conducting Software Engineering Surveys in China [66]	Junzhong_2008	Type I
The paper says: 1. "The sixth lesson learned was that people with sufficient software engineering knowledge need to be involved in the data collection process in order to respond to any confusion on the part of the respondents, and to ensure that the qualifications of the respondents are appropriate". 2. "Many project managers and developers do not have sufficient knowledge of software engineering methods and terms. It is necessary to supervise the entire data collection process, such as training people to assist in the survey, answering questions from respondents, and validating the quality of completed questionnaires".			

2.2.7 Conclusion

In the above table, the results of the literature review were categorized into 3 groups:

- The Type I group represents the reports that clearly claim that the participants' knowledge affects the survey results.
- The Type II group represents the reports that don't clearly claim whether the participants' knowledge affects the survey results or not.
- The Type III group represents the reports that clearly claim that the participants' knowledge doesn't affect the survey results.

The result presented in the above table show that the "impacts of participants' knowledge on the results of surveys" is controversial from the researchers' point of view. The papers with id K_1, K_7, K_9, K_{11} approve the impact, papers with id K_3, K_6, K_{10} reject the existence of the impact and the others K_2, K_4, K_5, K_8 paid attention to the importance of participants' knowledge but they have not check if there is a cause/effect on their results.

2.3 Conclusion of the literature review

Software quality evaluation as a field of software quality engineering is growing rapidly. The reports show that it is becoming more structured and is being taken into consideration more than ever. Software products are being used in many business areas from monitoring the governmental procedures to medical devices and so forth.

Although software quality standards are not yet as enforced as in other engineering disciplines; software practitioners take the advantage of *best practices* and quality *techniques* to deliver the software products at highest possible quality. In addition, to report the quality evaluation results, researchers use scientific techniques and tools such as statistical validation.

The reports of our literature review show that many researchers use "surveys" as the most popular tool for data collection in a quality evaluation project. It is also shown that almost all the published reports have followed similar steps for conducting the surveys. The steps presented by Kitchenham and Pfleeger (2013) have been used as a reference for many surveys in SE domain. However, concerns about 'who should participate' and 'who should not participate' in the surveys, are neglected in the literature.

From the reported list of the factors that influence the survey results, we addressed the participants' knowledge. The literature review shows that the "impacts of participants' knowledge on the results of surveys" is controversial from the researchers' point of view. Some researchers approve the impact, some reject the existence of the impact, and some do not take it into consideration.

2.4 Research motivation

The results of the literature review and the scoping review reveal that few papers are addressing the software quality evaluation from the users' point of view. In addition, capturing data for quality evaluation process is not conducted perfectly; several steps and validations are neglected. From those papers that address the software quality evaluation, few of them focus on the characteristics of the participants in the surveys while the researchers are willing to capture data. These lacks motivated us to investigate the influencing factors on software quality evaluation from the users' point of view.

CHAPTER 3 METHODOLOGY

As it was discussed before, our research objective is to create the profile of quality of a given software product. In Figure 8 the whole process for creating the quality profile is explained. The research process is composed of three phases: theoretical research, conducting the experiments, and analysis of the results. The research phase begins with creating the questionnaire, then questions are refined through a Delphi method and the answers of the participants are analyzed. The research ends up with creating the quality profile.

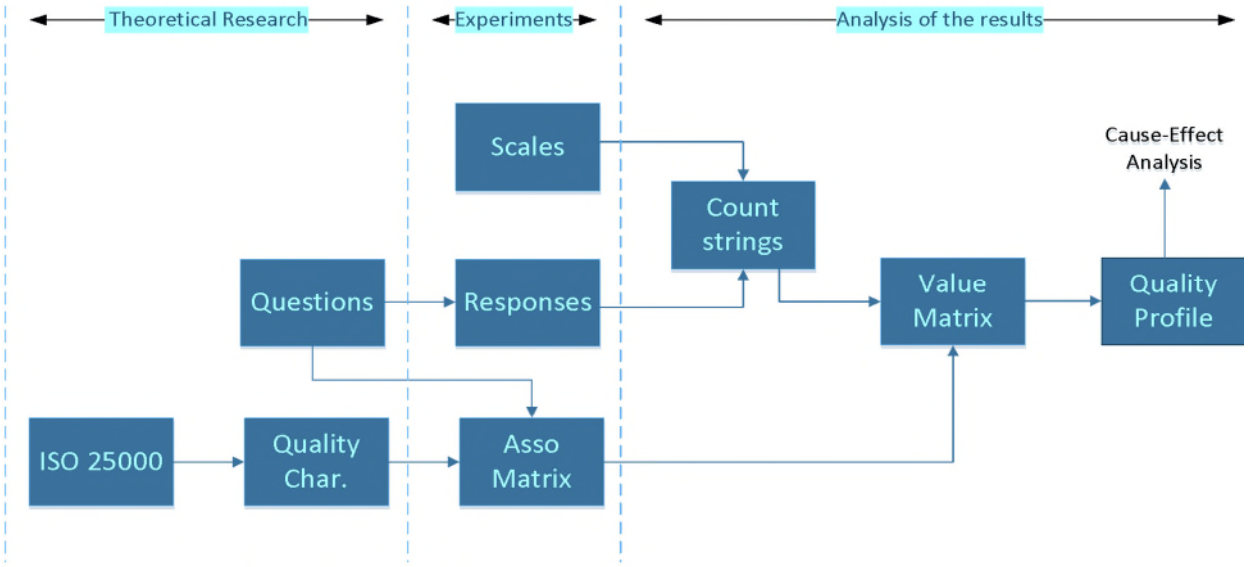


Figure 8: Research process to create the Quality Profile

3.1 Our quality evaluation model

In this study, the stakeholders' quality expectations are investigated and identified using a structured working questionnaire. The questionnaire consists of a number of questions regarding product quality characteristics. These characteristics are intended to be expressed in the stakeholders' language. The provided answers are utilized to recognize the most relevant quality characteristics and the associated values. In the following, the steps of creating, revising the questionnaire and finalizing the quality evaluation model are explained.

3.2 Questionnaire creation

At the initial step, a set of preliminary information in question form was elicited. In this step, using a free and unfiltered search, the researchers identified 152 questions related to quality measure elements. These preliminary questions were found by reviewing the related literature such as [75], [76], [2], websites, working instructions, norms, standards, and any other resources. It is worth mentioning that although this process is totally a non-systematic review, our methodology has the capacity to be initiated by any set of quality related questions. Since the questions will be refined in the next stages, they can be extracted non-systematically using a wild literature review, or using a systematic review for a specific context.

In addition to the questions, eleven different rating scales (Table 11) are also developed and assigned to each question.

Table 11: Rating Scales

Title	Scale
Absolute Yes/No	Absolutely No, Yes - for 20% to 50% of cases, Yes - for 50% to 80% of cases, Absolutely Yes
100% - 0% of time	75% - 100% of time, 50% - 75% of time, 25% - 50% of time, 0% - 25% of time
1h - 1w	Up to 1 hour, 1 hour to 24 hours, 1 day to 1 week, More than 1 week
High-Low Probability	High (70%-100%), Medium (30%-70%), Low (0%-30%)
All-None	None, Least, Most, All
Agree-Disagree	Disagree, Mostly agree, Completely agree
Importance	Absolutely Essential, Very Important, Of Average Importance, Of Little Importance, Not Important at All
Yes/No	Yes, No
Numeric	Number 1..10,000
Poor - Excellent	Poor, Fair, Good, Very Good, Excellent
Support	Unsupported, Poorly supported, Fairly supported, supported

3.3 Questionnaire refinement

In this step, the questions are refined using an iterative approach. The goal of the question refinement step is to make the questionnaire more reliable and valid. This step was done by adapting a Delphi method in three iterations. Delphi [2] is a systematic iterative communication technique that aims to increase the consensus among the members of a panel of experts. This step

was done by a panel of 7 experts in a software acquisition consultation company. The researcher acted as a coordinator. The expert team was asked to review the questions and determine if the questions and their rating scales are appropriate and relevant. The experts, who were selected by the company's chief technical officer, are staff with more than ten years of experience in software acquisition, enterprise software implementation, and deployment. The experts were also required to validate the questions by providing one of the pre-defined feedback types on the relevance of each of the questions.

As stated, the questionnaire was improved through an iterative approach. The purpose of the first iteration is to figure out how reliable the questions are, and how the ambiguities among the questions can be discovered and resolved. The reliability checking was done by asking the experts to review the questions. Accordingly, the questionnaire was conveyed to the panel of experts. The experts reviewed the questions and sent their feedback and comments to the coordinator. The coordinator applied the feedbacks to the questions and generated the composite report. The activities in the first iteration resulted in a reduction of 152 questions to 125.

The purpose of the second iteration is to find out how valid the modified questions are, and also to find out if the modifications in the first iteration have decreased the ambiguities and increased the convergence among the experts. For this purpose, six experts were asked to review the composite report and send their feedback and comments as Agree or Disagree to the coordinator. Thus, the main question that was asked of the experts in this iteration is "Do you agree with the modifications made for each question?" The items that were expected to be done by the experts in the second iteration are: 1) read the decision made for each question, 2) read the updated question and rating method, 3) read the comments of the other experts for each question, and 4) state your final opinion whether you agree or disagree with the final decision.

After receiving the feedbacks from the experts, we counted the Agreed and Disagreed answers and compared it with the obtained agreement value from the first iteration. The result shows that on average, the agreement value increased by 20%; i.e. from 58% to 78%. Applying the feedback also resulted in decreasing the number of questions from 125 to 106.

In the third iteration, the outlier questions were removed and also some questions were rephrased and polished to have only one type of rating scale for all the questions. The outliers are those questions that were not under agreement by the experts in the previous iteration. As it was

mentioned earlier, eleven rating scales were defined to categorize the questionnaire responses. These scales include both ordinal scales such as “Strongly disagree”, “Disagree”, “Agree”, “Strongly agree”, and absolute scales such as the integer values that are assigned to “number of help desk calls during one year of service”. For more clarification, three sample questions with their rating scales are presented in Table 12.

Table 12: Sample questions and rating scales

	Question	Rating Scale
Before Unification Scales	1. Is there a Help that helps, and matches the functionality?	Yes/No
	2. Are there features to distinguish mandatory fields?	Absolute Yes/No
	3. What proportion of potential users chooses to actually use the system?	All-None
After scales unification	1. The capability of the Help function within the product to provide adequate guidance on most issues.	Importance
	2. The capability of the product to present a feature to distinguish mandatory fields.	Importance
	3. The enjoyment of the users while using the product and feel fully engaged with it.	Importance

Comparing the data with different scales is problematic. To simplify this comparison, it is needed to unify all the various scales. For this purpose, we rephrased the questions to obtain one single scale that is “Importance”.

The other activity in the third iteration was removing the outliers from the questionnaire. We define the outliers as the controversial questions; i.e. the questions that were not approved by the majority of the experts. Consequently, we kept the questions that were approved by at least five experts (out of six) and removed the rest. By this way, 25 questions were removed, and 91 questions remained.

At the end of the third iteration, all the questions were approved by the industrial experts. Although the experts are all professionals in software acquisition, we asked two academic professors, who are professional in software quality and the user interface domain, to review the questionnaire. The goal of this complementary iteration is to review and validate the questionnaire from the theoretical, scientific, and academic point of view. The academic reviewers recommended removal of some marginal questions and only keep the ones that are focused on the software quality subjects.

By the result of this complementary iteration, we came up with 50 final questions. This set of questions is those that are validated and approved by both industrial and academic professionals.

In Table 13 a sample of the final questions for the end and the power users, and the shared questions for both groups are presented.

According to the comments received from the academic professionals, we separated the questionnaire for power users and end users. We define end users as the persons who regularly work with the software application for data entry and report generation. The power users are defined as the technical, knowledgeable professionals who have been granted high-level access rights for the software administration and control. Thus, the end users questionnaire includes the quality aspects closer to the quality in-use, and the power users' questionnaire consists of the questions more related to product quality defined in ISO/IEC 25000. The end user questionnaire includes 37 questions and power user questionnaire includes 33 questions. The end user's questionnaire is constructed from 17 questions specifically dedicated to the end users, plus 20 questions shared between the both user groups. The Power user's questionnaire is constructed from 13 questions specifically dedicated to Power users, plus 20 questions shared between the both user groups. In total 50 distinct questions are distributed in two questionnaires (17+20+13=50).

Table 13: Final Questions – sample

Group	Question
End User	Error handling agility: The capability of the product to handle data-input errors quickly and easily.
	Learnability: The capability of the product to be learned easily and quickly.
Power User	Tools for maintenance: The availability of the development tools for the technical power users to add features in the future.
	Stress Handling: The capability of the product to cope when exceeding various limits.
Both	Backward compatibility: The capability of the product to use the files and data created with older versions.
	Responsiveness: The capability of the software to perform most functions at the acceptable speed.

3.4 Association with the standard: A coding activity

The questions were associated with the quality measures by the researcher, and the quality measures were associated with the quality characteristics that are presented in ISO/IEC 25010. In this step, the concepts and the keywords of each question were retrieved, and then the questions were associated to at least one of the standard product quality characteristics. Since one question

may address more than one quality aspects, the relationship between the questions and the quality characteristics is a “many to many” relationship. This step resulted in creating a matrix of associations (Model matrix) with the questions in the rows and the quality characteristics in the columns. Each element of the Model matrix is marked (‘X’) if the given question in the row is associated with the quality characteristic in the column. In Table 14 a small part of a hypothetical Model matrix is presented. The marks in this matrix show that the question Q1 is associated with Effectiveness and Efficiency, the Q2 is associated with Efficiency and Satisfaction and Q3 is associated with Effectiveness and Satisfaction.

Table 14: Elements of a hypothetical Model matrix

	Effectiveness	**Efficiency**	**Satisfaction**
Q1	X	X	
Q2		X	X
Q3	X		X

As a sample, the diagram in Figure 9 shows the number of questions associated with the quality characteristics, for end users in quality in-use.

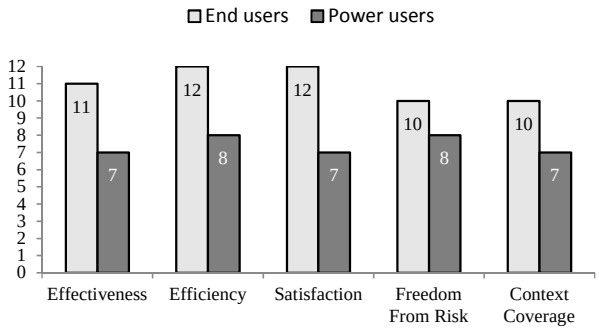


Figure 9: Number of questions associated with quality in-use characteristics for end users

All the questions in both end users’ and power users’ questionnaires are associated to 2, 3, or 4 quality characteristics as it is illustrated in Figure 10 The Y-axis in Figure 10 shows the question ids and the X-axis shows the number of quality characteristics associated with each question.

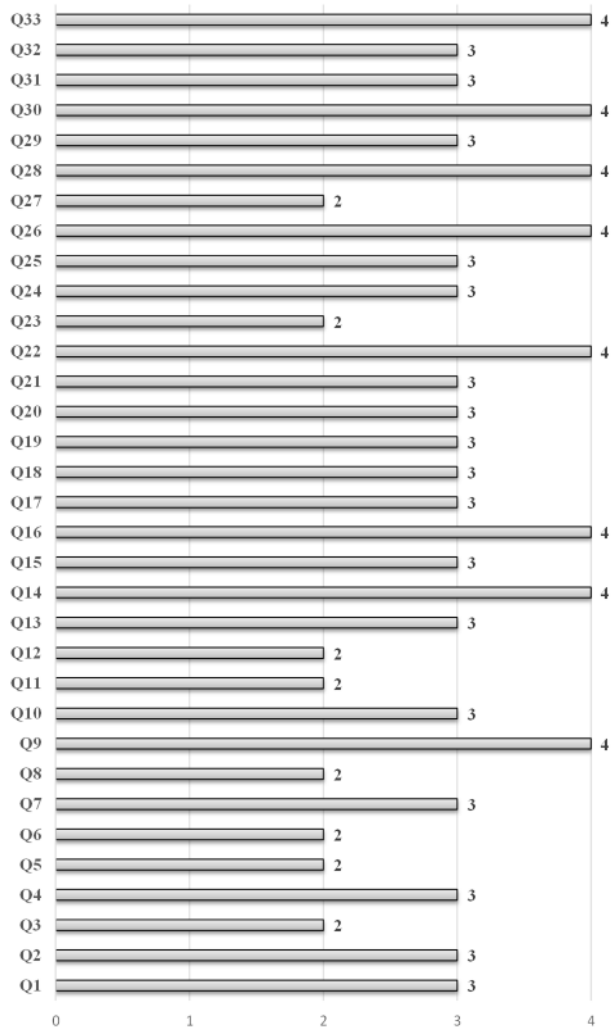


Figure 10: Questions and the number of associated quality characteristics

3.5 The case studies

To validate our methodology, we conducted two case studies. The first case includes a survey that aims to evaluate expected quality of a software which was under development at the end and power users' point of view. In the second case, we conducted two surveys to evaluate the observed quality of a software in operational mode from the end users' point of view. In the following sections, the methodology of the two case studies is explained.

3.5.1 Case #1:

In order to practically validate the quality evaluation model, we targeted the probable users of a software product. Using waterfall methodology, a development team attempts to create an administration and accounting software product for collegiate affairs. We applied our quality evaluation model to the software product by selecting a set of end users and power users and asking them to participate in a survey.

By asking 17 participants, including 9 end users and 8 power users, the survey was accomplished. The end users and the power users answered their respective questionnaire. The two questionnaires for the end and power user were implemented in the Google Forms application. Besides, an "Ethics Compliance Certificate" was also issued for this project. The anonymous responses were received and the questionnaires were closed after 15 days. In the following, the analysis process of the results is explained.

3.5.2 Case #2:

We conducted two surveys in a software consulting company named TEC[1] to evaluate the quality profile of the software they are using.

Since we have already published the details of the questionnaire creation and refinement procedure in previous papers [1][9], here we only focus on the results of the surveys to investigate if the participants' knowledge affects the results of the survey. Suffice to say, the questionnaire aims to create the product quality profile. Creating the profile includes eliciting the users' quality requirements, and then quantifying the elicited quality factors by applying them to the quality evaluation model and the ISO/IEC 25000 series of standards. The subject of the questionnaire can be any software product, which is deployed at any organization.

The objective of the surveys was to investigate the effects of participants' knowledge on the results of the questionnaires. The purpose was to evaluate the quality of Microsoft SharePoint (SP) at

[1]"TEC helps other companies to investigate, evaluate, and select the best enterprise software solutions for their unique business requirements. From small businesses to large enterprises, its clients include hundreds of private- and public-sector organizations in a variety of industries. A complete list of vendors and products serves the TEC software selection methodology supporting the software acquirers to make an efficient decision" [84].

TEC. The questionnaire includes 33 questions. In the 1st questionnaire, we asked the employees of Software Development department to participate. We call the participants of the 1st questionnaire as “tech-savvy” participants who are the employees familiar with software development technologies, programming languages, user interfaces, and common functions and feature of the most popular software applications. They are technically literate and spend almost all their daily time working with software applications. 33 employees were asked to participate, 17 employees responded (contribution rate = 50%)

In the 2nd questionnaire, we asked the rest of the TEC’s employees to participate (all the employees but the developers). We call the participants in the 2nd questionnaire as non-technical participants whose daily job doesn’t force them necessarily to work with almost any professional software applications. They are technically novices who occasionally work with general software applications. They spend daily a few minutes working with a software application. 110 employees were asked to participate, 30 of them do not use SharePoint at all in their daily jobs. 27 (out of 80) employees responded the questionnaire (contribution rate = 34%).

For each question, an identical set of Likert scales were provided. The scales were coded from A to E as in Table 15.

Table 15: The Likert scales

Code	**Likert Scale**
A	Fully supported
B	Mostly supported
C	Fairly supported
D	Poorly supported
E	Not supported
X	Don’t know

CHAPTER 4 SYNTHESIS OF PUBLICATIONS

During this PhD project, we accomplished three steps, each step was published in form of a paper. The papers present the related methodology of quality evaluation and the results that we obtained after conducting the experiments. A brief explanation of each paper and the obtained results are described in the following sections.

The first paper presents the questionnaire creation process. In this paper, we attempted to introduce our research objective as well as the steps that have been taken for collecting, purging and finalizing the list of quality related questions. This paper was accepted and presented at Software Engineering and Data Engineering Conference; SEDE 2015.

The second paper presents the methodology that we applied for creating the expected quality artifact. In this paper, the results of a case study that we conducted as a survey is also presented. Our results show how researchers can apply a standard-based quality evaluation project for retrieving the user's expectations. This paper was accepted and presented at IEMCON 2016 conference.

The third paper focuses on participants' knowledge as one of the influencing factors that impacts the survey results, as reported in the literature. The methodology and the results of our two case studies performed in a professional environment, for creating the observed quality artifact are also presented. The results show that the impact of participants' knowledge on the survey results is controversial among the researchers. This paper has been submitted to IST journal in 2017.

In the following, the three papers are discussed in detail.

4.1 Article 1: Industrial Validation of an Approach to Measure Software Quality

Preface: Software quality has emerged as a multi-faceted discipline that requires the software product to satisfy a wide range of stakeholders. Comprehensive specifications and evaluation of the quality of software are essential factors in ensuring value to stakeholders. Ensuring that a software product will add value to the stakeholders requires the software quality to be measured at specified milestones of the development process. The deviation between the expected and the

measured quality attributes of the final product shows how much of the stakeholders' requirements are satisfied. The main objective of this study is moving towards providing the quality deviation artifact through focusing on the differences between expected and observed quality. The rules and quality characteristics presented in ISO/IEC 25000 series of standard are applied to define a series of quality-related questions. The questions have been reviewed by the software evaluation experts in a company. The results show the importance of relying on a quality profile that is likely to outline the various perspectives of the stakeholder's concern by the quality of the product.

4.1.1 The results

The data for our study were obtained from the Technology Evaluation Centers (TEC) knowledgebase. As described earlier, the first step in creating a quality profile is requirement elicitation. In the current study, the researcher designed 151 questions related to quality measure elements. In addition to the questions, 10 different rating scales are also designed and are assigned to each question. We identified the data owners, who were the most likely appropriate for answering the various questions. A team of 7 experts working in TEC was asked to review the questions and determine if the questions and their rating scales are appropriate and relevant. The experts, who were selected by the company's CTO, are staff members with more than 10 years of experience in the field of software acquiring, enterprise implementation and deployment. The experts were also required to validate the questions by providing one of the 9 feedback types on the relevance of each of the questions. The feedback types which are listed in Table 8. The experts also could write a free text as comments for each question. The numbers of received comments are presented in Table 16. The results presented in Figure 11 shows that in average, 61% of the respondents found the questions and the rating scales as "Good".

Table 16: Number of comments per expert

Expert 1	**Expert 2**	**Expert 3**	**Expert 4**	**Expert 5**	**Expert 6**	**Expert 7**
45	25	42	56	35	111	49

This experiment was coordinated by the researcher. The experts, in isolation from one another, used their personal experience and judgment to evaluate the questions and send their feedback. The coordinator collected the feedbacks and prepared a summary of expert's feedbacks, which is called "composite report". At the end, each expert receives the composite report and was asked to make a new evaluation based on the feedback of others. This experiment is based on Delphi method that

is detailed in [2]. The obtained data can also be viewed from another perspective. The data presented in Table 17 shows the number of answers that are identical between the experts. For example, 85 options have been identically selected by the experts E3 and E5. The table shows that the expert E2 has the best agreement while the experts E6 and E7 have the least agreement value among the other experts.

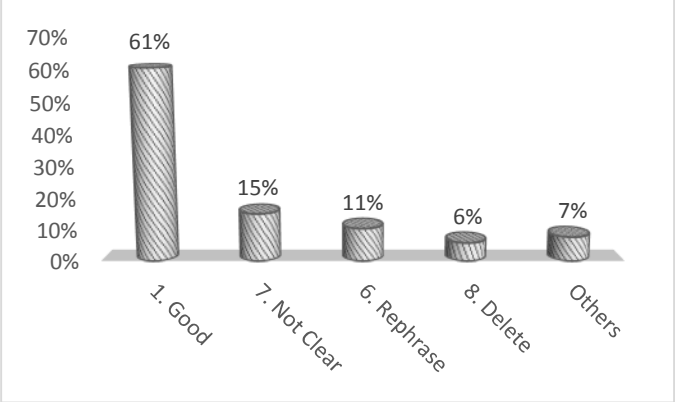


Figure 11: The average % of each received feedback type

Table 17 : Feedback Types

Option	Feedback description
1. Good	The question and its rating scale are appropriate
2. Change Rating Scale	The associated rating scale doesn't fit for this question
3. Split the question	The question can be decomposed into 2 questions as…
4. Combine	This question can be combined with the question number X
5. Duplicate->Remove	The question is duplicated with question number X
6. Rephrase	The phrasing is not appropriate. I suggest this "…"
7. Not Clear	It is not clear that what the question will measure
8. Delete	The question is irrelevant. It can't be answered.
9. Don't know	Out of my expertise

In addition to that, it can be seen that the agreement value between the experts E6 and E7 is the lowest value in the table. Although these two experts have almost the same working experiment

and academic education level, the results show that their viewpoints are different. Each question was revised based on the received feedbacks and comments. Since the “Not Clear” and “Rephrase” were the most selected options after “Good”, 63 questions were rephrased, and 40 rating scales were changed. Table 18 shows the number of modifications that were made for the whole questionnaire. Some questions had two modifications; consequently, the summation of the numbers in Figure 12 does not add up to 151.

Table 18: Number of identical feedbacks per Experts

	E1	**E2**	**E3**	**E4**	**E5**	**E6**	**E7**
E1		109	90	78	88	51	49
E2			99	91	91	50	51
E3				75	85	53	54
E4					80	52	52
E5						57	55
E6							48
E7							

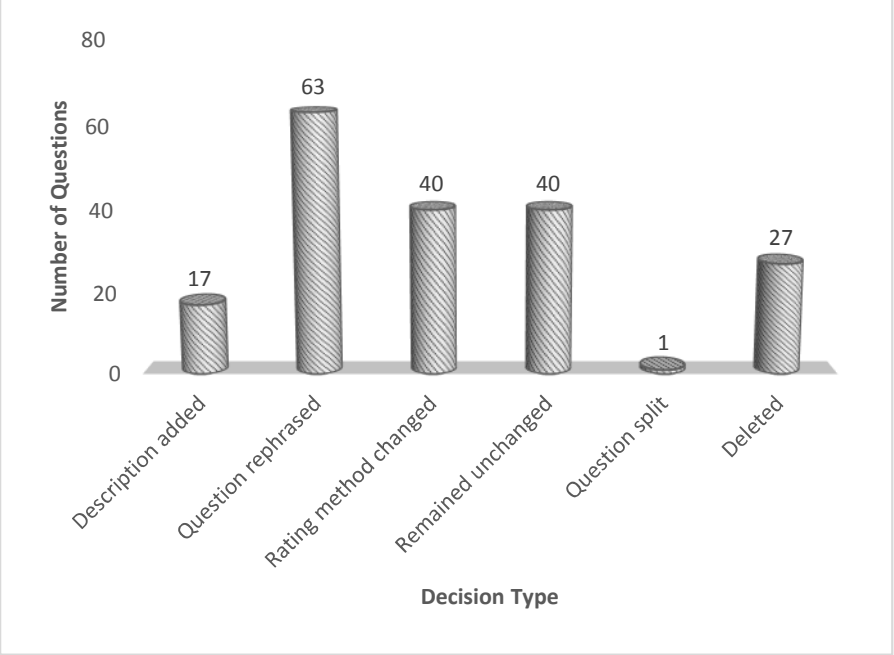


Figure 12: Number of questions for each decision

4.2 Article 2: Expected Software Quality Profile: A Methodology and a Case Study

Preface: For decades, the notion of software quality evaluation is raised as a challenging task. Recently many studies have presented quality evaluation methodologies for specific domains or specific techniques. They usually select a pre-defined model, customize the characteristics, define the metrics and evaluate the quality of the product or development process. Our study presents a bottom-up methodology for the quality evaluation process. In this paper, we present a methodology to create the expected quality profile. In our approach, the first step is listening to the users, and then retrieving the most important quality factors and creating a model to evaluate the expected quality of the software product. The profile is formed by eliciting the expected users' quality expectations, and then quantifying the elicited factors by applying them to our quality evaluation model and the ISO/IEC 25000 standard. The result of this research empowers the software development stakeholders to perform a crosscheck between users' specific quality expectations and other drivers (functional and architecture/design requirements), before or during the software development process. The crosscheck aims to guarantee that there are enough activities, roles and artifacts in the software development process to support the users' quality requirements.

4.2.1 The results

The analysis of nominal, interval and ratio scales seems to be straightforward and transparent while it is not for ordinal scales [77]. In a variety of applied fields, the level of measurement and the adequacy of treating ordinal data as interval data continues to be controversial [78] [79]. Thus, we applied a "counting" approach for analyzing the responses of the surveys. As the first step, the ordinal scales are identified from A to E, as the following:

A: Absolutely Essentials

B: Very Important

C: Of Average Importance

D: Of Little Importance

E: Not Important at All

Then the number of obtained scales for each question is counted, and a count_string is formed to show the number of each scale for a given question. The count_strings facilitate the counting and analysis of the responses. For example, the count_string=A2B5C2 means we have received 2 Absolutely Essential, 5 Very Important, and 2 Of Average Importance scales for a given question. In the next step, we put the count_string values in place of the marked elements of the Model matrix. Consequently, we will have another matrix called Value matrix. For example, if the count_string for the question Q1 is calculated as B3C4D3, for Q2 as A5D2E5 and for Q3 as B4D5E7, we have a hypothetical Value matrix for the above Model matrix, as it is illustrated in Table 19.

Table 19: Element of a hypothetical Value Matrix

	Effectiveness	**Efficiency**	**Satisfaction**
Q1	B3C4D3	B3C4D3	
Q2		A5D2E5	A5D2E5
Q3	B4D5E7		B4D5E7

At the end, the total of each scale of each quality characteristic is calculated. For instance, the data in Table 20 shows the number of each responded rating scale related to quality in-use for end users.

Table 20: Numbers and percentage of each rating scales: End user questionnaire – Quality in-use

	Effectiveness	**Efficiency**	**Satisfaction**	**Freedom from Risk**	**Context Coverage**
↓ Five-Scale ↓					
A	31(33%)	47(36%)	26(20%)	24(31%)	12(18%)
B	32(34%)	45(34%)	43(33%)	32(42%)	19(28%)
C	25(27%)	29(22%)	44(34%)	12(16%)	24(36%)
D	2(2%)	4(3%)	6(5%)	4(5%)	2(3%)
E	4(4%)	6(55)	12(9%)	5(6%)	10(15%)
↓ Three-Scale ↓					
A+B	67%	70%	53%	72%	46%
C	27%	22%	34%	16%	36%
D+E	6%	8%	14%	12%	18%
Sum	100%	100%	100%	100%	100%

As it was mentioned earlier, 20 questions were shared in the two questionnaires for the end and the power users. We applied the Fisher's Exact Test to check if there is a significant difference between the end and power user's responses. The results of the Fisher's Test are shown in Table 21 . We defined the null hypothesis as "there is no significant difference between the two categories".

Table 21: The p-value resulting from Fisher's test

Question ID	p-value		Question ID	p-value
Q1	1		Q11	0.61
Q2	1		Q12	1
Q3	1		Q13	0.52
Q4	0.55		Q14	0.26
Q5	0.78		Q15	0.26
Q6	1		Q16	0.46
Q7	1		Q17	0.64
Q8	0.87		Q18	0.46
Q9	1		Q19	0.40
Q10	0.43		Q20	0.82

Fisher's test calculates the p-value, to show if the null hypothesis can be rejected. If the p-value is less than 0.05 then the null hypothesis can be rejected, otherwise, the alternative hypothesis can be accepted. The value of p-value was calculated in R. The Fisher's tests are applied to the shared questions. Because the p-values are more than 0.05, we cannot reject the null hypothesis. In other words, we can say that the values are from the same distribution. This means that both end users and power users have the same belief about the importance of the certain quality factors, by using our questionnaires. In other words, the results of all p-values strongly show that there is no significant difference between the quality expectations of the both user groups.

- Expected quality profile

We designed the expected quality profile as a three-scale percentage style diagram. We compressed the five-scale presentation to three-scale by:

Very Important = "Absolutely essential" + "Very important"

Neutral = "Of average importance"

Not Important = "Of Little Importance" + "Not Important at All"

The data of the expected quality profile for the end users in quality in-use characteristics are shown in Figure 13. The percentage stacks show that from the end users point of view, the Effectiveness, the Efficiency, and the Freedom from Risk are the most important quality characteristics, while they relatively find the Context Coverage a less important factor.

The similar profile can be created for product quality characteristics, as it is shown in Figure 8. The profile in Figure 14 includes the percentages of importance for the product quality characteristics of end users. The same diagram can be created for the power users. The percentage stacks shown in Figure 14 implies that the Security and the Reliability are the most important characteristics from the end users' point of view while the Portability is the least important factor.

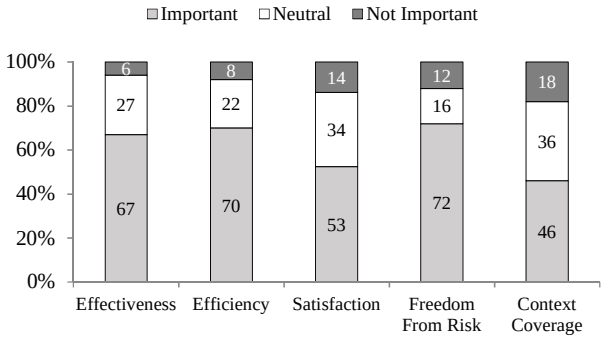


Figure 13: Expected quality profile - End users - Quality in-use

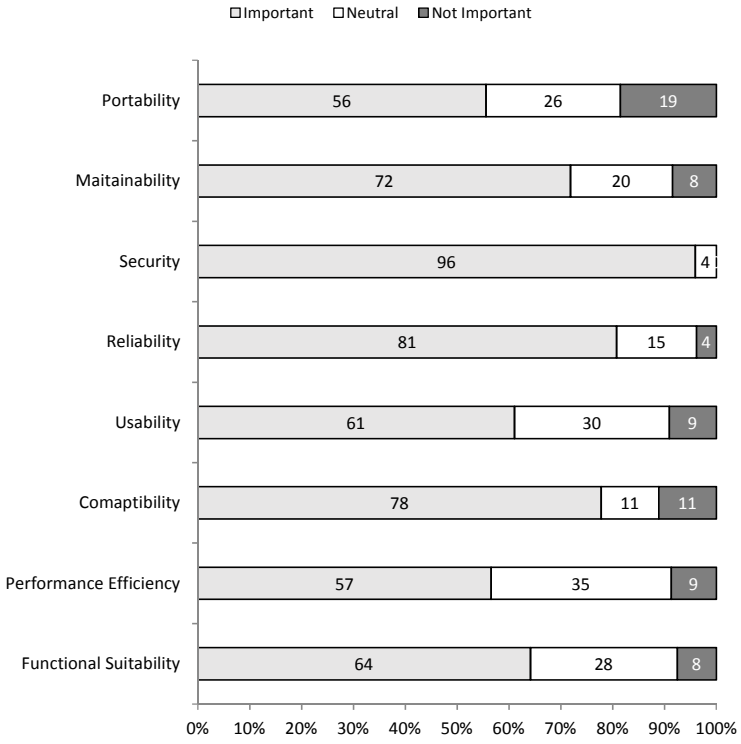


Figure 14: Expected Quality profile – End users - Product quality

- Cause-effect analysis

The creation process of our quality evaluation model is a round trip movement that includes two stages: 1) forward construction that is a macro perspective, and 2) backward analysis that is a micro perspective. Until now, we constructed the questionnaires, gathered the data from the real users and created the high-level quality profiles. The next stage is to move backward to the details and see what causes the importance values shown on the profile diagrams. For answering this question, we need to do a filtration on the data addressing the questions that cause the profile. The tool that we used for this illustration is the Pareto analysis. The Pareto analysis is a technique that shows the percentage of the influencing factors that causes the final result.

For example, on the end users' quality profile, the importance value of the "Satisfaction" is 53% as shown in Figure 13. It would be helpful if we could know "what are the questions that caused the majority portion of the 53%". The Pareto analysis gives the answer. As it is illustrated in the Pareto chart in Figure 15, the question number Q30, Q19, Q20, Q16, Q29, Q8, Q33, Q32 have formed the basis for 84% of the importance. Thus, we can say if the software is able to answer the above questions, then it is able to "satisfy" 84% of the end user's quality expectations. The Pareto analysis can be applied to other quality characteristics on the profile for end users and power users. It should be noted that the "84%" is chosen only as an example. Any other value can be chosen as the boundary of the cause-effect analysis.

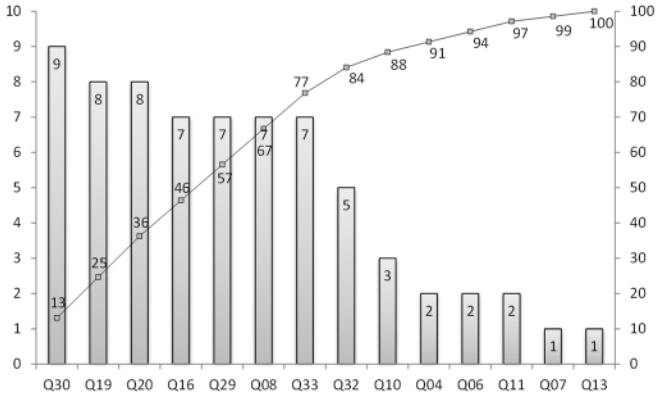


Figure 15: Pareto analysis for "Satisfaction" – End users

4.3 Article 3: Does Participants' Knowledge Affect the Survey Results? A Systematic Literature Review in Software Engineering Domain

Preface: Along with the growth of the studies in software engineering domain, questionnaire-based surveys have also been of interest to researchers. There are many factors influencing the results of the questionnaires, such as questions wording, questionnaire formatting, the respondents, the generalizability of the results, etc. In this research, we focused on the effects of the participants' knowledge on the survey's results since it is rarely addressed in the literature, while it can affect the validity of the survey. We performed a systematic literature review to reveal that how the researchers in software engineering are conducting the surveys and what factors they are neglecting. In addition, we conducted two surveys in a software consulting company to check if the participants' knowledge affects the survey results. Our systematic literature review shows that the existence of this effect is controversial among the researchers. We also presented the results of our two surveys in this paper. Our surveys approve that the participants' knowledge may affect the result of the surveys. The audience of this paper is the researchers who will manage a survey and/or evaluate survey results.

4.3.1 The results

The whole data show that the respondents in the 2nd questionnaire have selected more Don't Know (X) options, than the 1st questionnaire respondents. In other words, the non-technical participants (in the 2nd questionnaire) have less knowledge about the SharePoint quality factors at TEC than the technical users (in the 1st questionnaire). The chart in Figure 16 shows the results in more details. These data are normalized.

During the data analysis process of the questionnaire, the answers were associated with the standard quality characteristics. Then they were categorized in three scales; 'supported', 'so-so', and 'not supported', as follow:

$$Supported = A + B = [Fully\ Supported] + [Mostly\ Supported] \quad (1)$$

$$So\ So = C = [Fairly\ Supported] \quad (2)$$

$$Not\ Supported = D + E = [Poorly\ Supported] + [Not\ Supported] \quad (3)$$

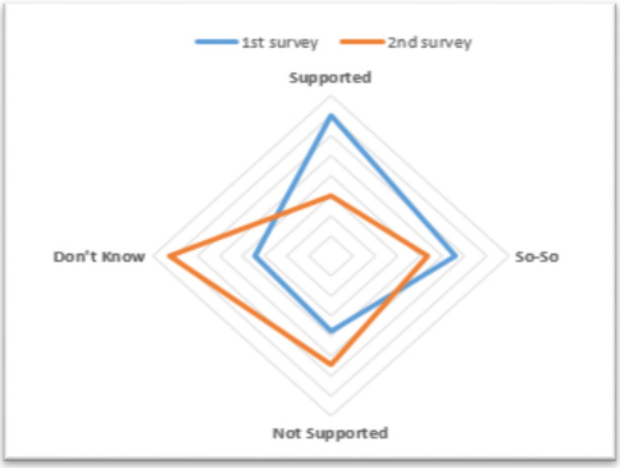


Figure 16: The scales in both surveys

The charts in Figure 17, Figure 18, Figure 19, and Figure 20 show the details of the breakdown distribution of the resulted quality characteristics. These charts illustrate that the respondents in the 1st questionnaire (who are the technical users) believe that SharePoint at TEC supports more quality characteristics than the respondents in the 2nd questionnaire (who are non-technical users). To have a set of valid data for comparison, all the comparative data have been normalized. Table 22 shows a real example of data before and after the normalization process. The raw data are retrieved directly from counting the answers and then they have been normalized to 0-100 range. For example, according to Table 22, for Effectiveness, we have received 12 ‘Fully Supported’ answers. It means that 4% of all the answers received for all the scales for Effectiveness belong to ‘Fully Supported’. The same argument shows that 42% of the answers belong to ‘Don’t know’.

Table 22: The scales and their normalized values for Effectiveness in the 2nd questionnaire

Effectiveness		
Scale	***Raw***	***Normalized***
Fully Supported	12	4%
Mostly Supported	19	7%
Fairly Supported	62	22%
Poorly Supported	53	19%
Not Supported	16	6%
Don't know	116	42%
TOTAL	**278**	**100%**

Consequently, the vertical axes in the following radar charts are in 0-100 range. For example, the chart titled ‘Don’t Know – Product Quality’ shows that 30% of the Don’t Know answers from the 1st questionnaire is related to Functional Suitability, while this value is 40% in the 2nd questionnaire.

Figure 17: Quality Profiles for 'Supported' scale

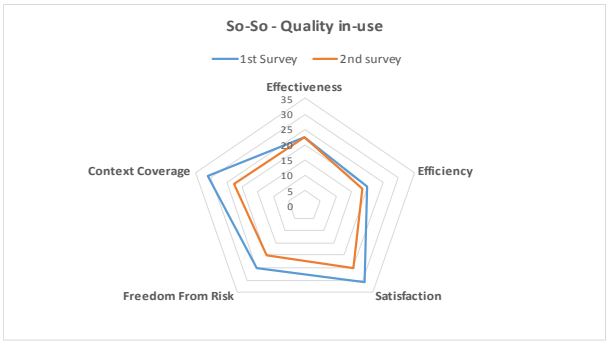


Figure 18: Quality Profiles for ‘So-So’ scale

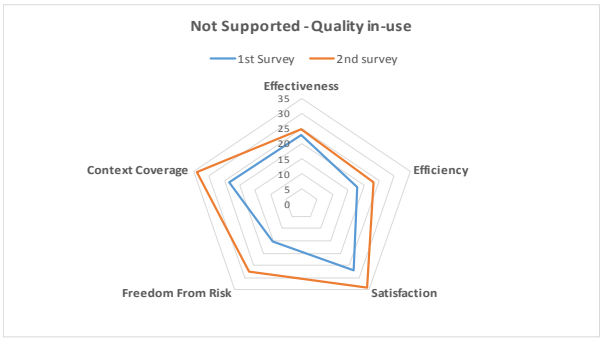


Figure 19: Quality Profiles for ‘Not Supported’ scale

Figure 20: Quality Profiles for 'Don't know' answers

The last two charts (titled: Don't Know) represent the lack of technical knowledge of the respondents in both questionnaires. However, the charts show that the number of X's has been increased in the 2nd questionnaire in both quality in-use and product quality related characteristics.

To show how the results of the two questionnaires are distributed, we performed a statistical analysis using Fisher's Exact Test. For this purpose, we compared the five-scale percentages of the 1st and the 2nd questionnaires for each quality characteristic. The Null Hypothesis is: "The participants in both questionnaires have the same belief about the total quality of SharePoint at TEC". If p-value > 5% then we cannot reject the null hypothesis. If p-value < 5% then we reject the null hypothesis.

Table 23 shows a sample of our data. The p-value is calculated for $(\boldsymbol{A_1}, \boldsymbol{A_2})$ columns, as well as for $(\boldsymbol{B_1}, \boldsymbol{B_2})$, and so on. Each two columns represent the data of a specific standard quality characteristic. In the most left column, the rows represent the rating scales.

Table 23: p-values obtained from Fisher's Test for three sample quality characteristics of both questionnaires

	Effectiveness		**Efficiency**		**Satisfaction**		Other values for other quality characteristics…
	1st	**2nd**	**1st**	**2nd**	**1st**	**2nd**	
	A1	A2	B1	B2	C1	C2	
Fully Supported	8	4	11	7	5	3	
Mostly Supported	18	7	20	10	19	6	
Fairly Supported	22	22	20	18	25	25	
Poorly Supported	20	19	16	20	13	31	
Not Supported	3	6	2	4	3	4	
Don't know	29	42	30	41	31	31	
Fisher's p-value	**9%**		**21%**		**1%**		

The p-values are shown in Table 24 for each quality characteristic.

Table 24: the Quality Characteristics and the p-values

Quality in-use	**Fisher p-value**
Effectiveness	9%
Efficiency	21%
Satisfaction	1%
Freedom from Risk	7%
Context Coverage	2%

Product Quality	**Fisher p-value**
Functional Suitability	5%
Performance efficiency	0%
Compatibility	0%
Usability	1%
Reliability	51%
Security	67%
Maintainability	0%
Portability	27%

Since the results show that in almost 50% of the quality characteristics the p-value < 5%, so we cannot accept the null hypothesis. We can conclude that the participants in the both questionnaires have different beliefs about the quality of SharePoint at TEC. We have already argued that the number of X's have increased in the 2nd questionnaire (non-technical users). Now the results of the fisher's test show that the two groups of the participants are somehow thinking differently about certain quality aspects of the software under study. The data in Table 25 show the questions and the number of Don't Know items (identified as X) in both questionnaires.

Table 25: Number of X's for each question in the both questionnaires

Questions	**Number of X's (Questionnaires)**	
	1st	**2nd**
1- Undo availability: The consistent availability of an 'Undo' feature, even after writing to the database.	5	13
2- Audible warning: The capability of the product to present audible warning.	13	17
3- Help usability: The ability to the Help function to provide adequate guidance on most issues.	4	11
4- Error-message coherence: Are the product's error messages clear and informative?	2	7
5- Clarity in user interfaces: Does the product provide clear, explicit and well-worded language within the user interfaces?	0	2
6- Mandatory fields: The product's ability to present a feature that distinguishes mandatory fields from non-mandatory ones.	1	4
7- Screen traceability: The product's ability to show where users are in the application on any/every screen.	1	6
8- Default values: The product's ability to allow users to define default values for desired fields using algorithms, equations, or business rules.	7	15
9- Screen color customization: The product's ability to allow users to adjust screen colors.	9	19

Table 25: Number of X's for each question in the both questionnaires

Questions	**Number of X's (Questionnaires)**	
	1st	**2nd**
10- Minimalism: Does the product avoid redundant content or repeated feature entry-points?	2	10
11- Report customization: The product's ability to sort, filter, and customize standard reports.	7	11
12- Documentation accuracy: The product's ability to present the correct and complete user- and technical documentation.	8	19
13- Function key availability: The availability of function keys for frequently used menu items, or frequently used data entries.	7	18
14- Field word processing: The availability of advanced word processing functionality within alpha fields.	6	15
15- Learnability: How easily and quickly learned the product is for most users.	0	7
16- Controllability: The user's feeling of control over the proceedings of the software.	0	8
17- Secure remote accessibility: The ability of the product to be accessible remotely via a secure connection.	7	11
18- Mobile Accessibility: The ability to access the product with mobile devices such as smartphones, tablets, etc.	14	18
19- Error handling agility: The product's ability to handle data-input errors quickly and easily.	6	9
20- Responsiveness: The product's ability to perform most functions at an acceptable speed.	0	3
21- Satisfaction and interest level: The enjoyment or satisfaction of the users while using the product, and how engaged they feel with the product.	0	0

Table 25: Number of X's for each question in the both questionnaires

Questions	**Number of X's (Questionnaires)**	
	1st	**2nd**
22- Expectancy: The product's ability to exceed expectations and meet needs that the user did not know he/she had.	2	3
23- Import/export options: The possibility to import/export data into the software using any type of file format.	8	17
24- Controlled modifiability: The product's capability to be "stable" and "reliable".	0	7
25- Backward compatibility: The product's ability to use files and data that were created with older versions of the product.	8	20
26- Interoperability: The ability of the product's components to interact with each other without undue delays or problems.	6	15
27- Authentication: The product's ability to identify users through log-in or other means.	2	6
28- Authorization: The product's ability to follow authorization protocols in allowing users to achieve their authorized level of access and use.	6	8
29- Privacy: The product's ability to protect data from being seen by unauthorized users.	9	13
30- Compliance: The product's ability to adhere to standards, conventions, or regulations in laws and similar prescriptions.	10	22
31-Access rights: The product's ability to assign different rights per user types in different sites.	9	13
32- Concurrency: The product's ability to perform multiple tasks in parallel without delays or problems.	5	11
33- Disaster Recovery: The product's ability to recover from an unexpected failure without losing user information.	9	18

Although this table shows the number of X's, it is more illustrative and expressive if we use the percentages of Xs instead of the number of Xs. Therefore, for each question, we calculate the percentages as:

$$Percentage\ of\ X\ for\ Question\ Q_i = \frac{Number\ of\ X's\ received\ for\ Q_i}{\sum A, B, C, D, E, X} \qquad (4)$$

CHAPTER 5 GENERAL DISCUSSION

During the data analysis, especially in case #2, we found out that there is a number of Don't Knows answered by the participants. Thus, it led us to deeply analyze the 'don't know' and check if there is a pattern that justifies this amount of don't know. In this chapter, we first discuss the systematic method that we applied to remove the outliers from our data set, and then we discuss the obtained don't know. At the end of this chapter, we deal with the risks that threat the validity of our research.

5.1 Systematic outlier removal

To validate our data set, it is required to identify the outliers – or false data points based on the execution of the exploration project, such as checking if the participants have participated seriously in the survey [39]. For this purpose, we applied a systematic method.

First, we present all the questions and the number of Xs. The numbers in Table 26 show the percentages of 'don't know' (which is denoted as X in this thesis) over the total number of the received answers for each scale; from A to X. For example, for the question #1, 29% of the answers in the 1st questionnaire, were X. This value is changed 46% in the 2nd questionnaire.

Table 26: The percentage of Xs received for each question in both questionnaires. The * denotes the outliers

Question #	**1st questionnaire (%)**	**2nd questionnaire (%)**	**Question #**	**1st questionnaire (%)**	**2nd questionnaire (%)**
1	29	46	18*	82	64
2*	76	61	19	35	32
3	24	41	20	0	11
4	12	25	21	0	0
5	0	7	22	12	11
6	6	14	23	47	61
7	6	21	24	0	26
8	41	54	25	47	71
9*	53	68	26	35	54
10	12	37	27	12	21
11	41	39	28	35	29
12	47	68	29	53	46
13	41	64	30*	59	79
14	35	54	31	53	46
15	0	26	32	29	39
16	0	30	33*	53	64
17	41	39			

Using this analysis method, we can identify the outliers systematically. For example, if we set the threshold to 40%, it means that the questions are considered outliers when the number of X's are more than 40% of the total number of received answers. This rule is set for both questionnaires. Thus, there are 12 outliers for threshold=40%. We can control the threshold to get the optimum number of outliers. Obviously, the higher the threshold values the lower the number of outliers. That is:

$$Threshold \propto \frac{1}{Number\ of\ outliers} \quad (5)$$

Since the value of the threshold is used to clean up both questionnaires from the outliers, selecting the optimum threshold value should be done with care. One the one hand, if the threshold is too high, then a lower number of outliers are identified, and many unrelated and vague questions, that result in X answers will remain in the questionnaire. On the other hand, if the threshold is too low, then higher number of outliers are identified, and many valuable and meaningful questions are removed from the questionnaire. In both cases, an inappropriate value of thresholds negatively affects the accuracy of the questionnaire. Table 27 shows the relationship between the threshold and the number of outliers.

Table 27: The identified outlier questions based on the given thresholds

Threshold %	Number of outliers
40	12
45	10
50	5
55	3
60	2

It seems that the threshold=50 is good enough since it gives 5 outliers; neither too high nor too low. If we look into the questions that will be removed by threshold=50%, we find out that those questions are somehow beyond the knowledge of end users in an organization. Thus, removing them from the questionnaires does not affect the accuracy of the questionnaire. Threshold=50 means that if we ignore the questions that at least 50% of the answers are X, then we can consider 5 questions as an outlier and remove them from the list of questions.

Table 28, provides the explanation for the reason why the participants do not know about the above questions identified as outliers. We categorize the reasons as:

- Functionality not needed
- Features not possible
- Don’t care
- Not happened so far
- Lack of knowledge

Table 28 : Justifications for outliers

Outlier item	Reason	Explanation
2 - Audible warning	• Not possible	Since making the noise is not allowed in the office during the working hours, there is no speaker connected to the computers. So, the feature – if exists – would not be usable for them. That's why most of the users don't know about it.
9 - Screen color customization	• Not needed	Most of the users do not know if they can change the color of SharePoint screen items because each user works with a limited number of SharePoint items. So, it is not necessary to highlight or mark a specific item in the SharePoint pages using colors. This feature is not useful for the TEC users so they do not know about it.
18 - Mobile accessibility	• Don't care • Not needed	The users that need to connect to the SharePoint remotely, can do that because they can connect to their machines remotely and then they will have all the installed applications such as SharePoint, and so on. So, they do not need to know if SharePoint has a capability to be accessed remotely on the mobile devices. Other users do not need to connect remotely.
30 - Compliance	• Do not care	Only a few number of the employees are involved in defining, analyzing and revising the TEC's business processes and to make sure that they comply with global best practice, norms, and standards. Most the users are just the end users who only work on final forms and features. So, it is expected that assuring the compliance with the standards is not followed by most of the users and therefore they do not care about it.

Table 28 : Justifications for outliers

Outlier item	Reason	Explanation
33 - Disaster recovery	• Not happened so far • Lack of knowledge	Many users have not any experience on any SharePoint disaster at TEC. Therefore, because it is not occurred until now, they have no knowledge about the reaction of SharePoint at this case.

We removed the outlier questions from the list and then analyzed the results excluding the outliers. The numbers and the charts hereinafter are based on the refined data, i.e. all the data excluding the outliers.

5.2 Conclusion

Looking at the data from a higher level of abstraction, and the results of the Fisher's exact test revealed that different user groups have a different understanding about the quality of a specific software application. In our case, the users in development department who are known as technically knowledgeable users find the software as higher quality than the users in other departments of the company. The key point here is the "knowledge". Therefore, we can say: "the software quality from the users' point of view depends on the knowledge of the users about the software developments and quality, in general, and on the application under study, specifically". In our case, the skilled users in software technical departments claimed that the SharePoint has higher quality, than the other users working in other departments.

5.3 Where the knowledge lacks

In the next step, we focus on the missing knowledge of the users in the 2nd questionnaire of the 2nd case study. For this purpose, we define three indicators that enable us to select the departments and the users who suffer from lack of knowledge. Then we retrieve the questions - i.e. quality measure elements - that are related to the users' missing knowledge.

5.4 The indicators

In this stage, we attempt to answer to question: “what is the missing Knowledge in the 2nd group of participants that causes them to select more X’s than the 1st group?”. This question led us to analyze the Don’t Know answers deeply among the 2nd questionnaire answers. In these analyses, the goal is to focus on the users that use SharePoint as one of their main daily tasks, while they have still ambiguities about SharePoint. Then we investigate on SharePoint quality elements that are missing for those users. For this purpose, we defined three indicators as follow.

1. **SharePoint Involvement Percentage per TEC Department (SPIPD):** SharePoint is used to manage the business processes at TEC. At the time of conducting our surveys, 17 TEC’s business processes had been implemented in SharePoint. Given that we have a list of the 17 processes, and we know how the TEC departments are involved in certain processes, we define an indicator named “SharePoint Involvement Percentage” (SPIP). This indicator is aimed to illustrate how the TEC’s departments are involved in SharePoint.

For department D, the SPIP is defined as:

$$SPIP_D = \frac{\sum processes\ that\ department\ D\ is\ involved\ in}{\sum processes\ available\ in\ SP} \times 100 \qquad (6)$$

For example, for the *Marketing* department which is involved in 10 SharePoint processes, the involvement is 59%:

$$SPIP_{Marketing} = \frac{\sum processes\ that\ Marketing\ is\ involved\ in}{\sum processes\ available\ in\ SP} \times 100 = \frac{10}{17} \times 100 = 59\%$$

That means in average, the Marketing department is involved in 59% of the TEC’s processes. The data in Table 29 shows the SPIP values for all 12 departments. It should be noted that, since some departments have more than one process, we have 12 departments but 17 processes.

Table 29: SharePoint Involvement Percentage (SPIP)

Department	$SPIP_D$	Department	$SPIP_D$
Testing	41%	Project Development	59%
Editing/Translation	53%	R&D	41%
Electronic data interchange	29%	Research Analyst	53%
Executive	41%	Selection Services	29%
Marketing	59%	Small business groups	29%
Pre-Sales	41%	Vendor Services	29%

2. **Applicability of the Questions (APL_Q):** not all the questions are applicable to all the departments. The APL_Q is a percentage that indicates the relationship between the questions and the departments. The data for associating the questions to TEC departments are collected from various interviews: with department employees, with SharePoint administration team, department documents and reviewing the recorded tasks in SharePoint. Table 30 shows the associations between the questions and the departments; the letter 'Y' denotes an association between the given departments in the row with the given question in the column. Because of the lack of space in Table 30, only four sample questions denoted as Q_1 to Q_4 are presented.

Table 30: Association of four sample questions with the departments.

Y=Associated N=Not

Department	**Q_1**	**Q_2**	**Q_3**	**Q_4**
Testing	Y	N	Y	Y
Editing/Translation	Y	Y	Y	Y
EDI	Y	Y	Y	Y
Executive	Y	N	Y	Y
Marketing	Y	N	Y	Y
Pre-Sales	N	Y	Y	Y
Project Development	Y	Y	Y	Y
R&D	Y	Y	Y	Y
Research Analyst	Y	Y	Y	Y
Selection Services	Y	Y	Y	Y
Small business groups	N	Y	Y	Y
Vendor Services	N	Y	Y	Y
APL_Q (%)	**75**	**75**	**100**	**100**

Table 31 presents the value of APL_Q for all the questions. The values show that there are some general questions that are related to all departments (such 7575as Q_3, Q_4, Q_5, etc.), and questions that are applicable for some of the departments only. The last row shows the values of APL_Q. For example, the value of 75% for the question #1 indicates that this question is applicable for 75% of the TEC's departments.

Table 31: Applicability of the questions for the departments

Question #	APL_Q	Question #	APL_Q	Question #	APL_Q
1	75	**10**	17	**19**	100
2	75	**11**	25	**20**	33
3	100	**12**	58	**21**	100
4	100	**13**	100	**22**	50
5	100	**14**	100	**23**	33
6	67	**15**	83	**24**	100
7	75	**16**	100	**25**	100
8	92	**17**	100	**26**	50
9	67	**18**	100	**27**	58
				28	33

Another indicator is defined to show how a specific department is involved in the questions. We call this indicator as INV_D, which is explained below.

3. **Involved departments (INV_D):** The values for this indicator can be extracted by looking at the department question associations from the reverse point of view. The INV_D indicates a percentage of the questions that the department *D* is involved in. For example, the department '*Small Business Group (SBG)*' is involved in 50% of the questions:

$$INV_{SBG} = \frac{Number\ of\ questions\ that\ SBG\ department\ is\ involved\ in}{Total\ questions} = \frac{14}{28} \times 100 = 50\% \quad (7)$$

Table 32 shows the INV values for all departments.

Table 32: Involved Departments

Department	INV_D	Department	INV_D
Testing	93	Project Development	89
Editing/Translation	82	R&D	96
EDI	57	Research Analyst	75
Executive	82	Selection Services	61
Marketing	86	Small business groups	50
Pre-Sales	68	Vendor Services	57

5.5 Question Applicability vs. don't know

The goal here is to retrieve the questions that are applicable to many departments and at the same time, many users do not have knowledge about the quality element that lies in that question. It means we highlight the questions that both value of 'APL_Q' and the 'count of X' are high. However, we pay attention to two different types of X: 1) the answer X's which are applicable to the department, and 2) the answer X's that are not applicable. For this purpose, we created a matrix named Applicability-Data Matrix by combining the following two matrixes:

- The Applicability Matrix that shows if a specific question is related to a specific department. The rows of this matrix are the departments, the columns are the questions, and the elements are 'Yes' or 'No'. in this matrix, 'Yes' means that the given question is applicable for the given department, and 'No' means the given question is irrelevant for the given department.
- The Data Matrix that includes the answers of the users to the questionnaire. The rows of this matrix are the participants, the columns are the questions, and the element can be one of A, B, C, D, E, or X.

Thus, the rows of the Applicability-Data matrix are the departments, the columns are the questions and the elements will be pairs of (P_1, P_2), where:

$$P_1 = \{A, B, C, D, E, X\} \qquad P_2 = \{Yes, No\} \qquad (8)$$

So, the format of the elements of the Applicability-Data matrix is defined as:

$$(P_1, P_2) = (response, applicability) \qquad (9)$$

This process was done for all the user answers. Table 33 shows the Applicability-Data matrix for a subset of departments, users, and questions. In this table, the users are coded as letter 'U' and a subscript number that shows the code of the user. The other letters and the Yes, No are explained in the equations (8).

Table 33: The Applicability-Data Matrix

Departments	**Users**	**Q_1**	**Q_2**	**Q_3**	**Q_4**	
Testing	U_{21}	D Yes	C No	B Yes	C Yes	
Testing	U_{27}	E Yes	C No	D Yes	C Yes	
Editing/Translation	U_{16}	X Yes	X Yes	B Yes	D Yes	
Editing/Translation	U_{20}	D Yes	D Yes	D Yes	C Yes	
Electronic Data Interchange	U_7	C Yes	D No	C Yes	C Yes	
Electronic Data Interchange	U_8	C Yes	D No	D Yes	C Yes	
Electronic Data Interchange	U_{25}	X Yes	D No	D Yes	D Yes	
Executive	U_9	E Yes	E No	D Yes	D Yes	
Marketing	U_6	E Yes	X Yes	C Yes	B Yes	
Marketing	U_{15}	E Yes	E Yes	D Yes	C Yes	
Pre-Sale	U_4	X No	X Yes	X Yes	C Yes	
The rest of the rows of the matrix…						

The rest of the columns of the matrix…

The matrix in Table 33shows that how the user U_i who belongs to the department D_d has answered to the questions Q_q, and if U_i should have the knowledge of the question Q_q. for example The 'X Yes' value of U_6 shows that he does not know about the point of the question Q_2, while the Q_2 is applicable for Marketing department and the user U_6 in the same way.

The results also show a strong negative correlation between the values of applicability (APL_Q) and the number of X of the questions (-0.80). This correlation can be interpreted as "the more the

questions are applicable for the users, the less X we have received". In the other words, the users that have fewer ambiguities about the quality elements that are applicable to them; their missing knowledge is mostly about the items that are not related to their daily tasks.

5.6 The knowledge missing items

Now we have enough data to analyze the questions that users do not know about them while they are relevant. We label the 'knowledge missing' questions as the ones that we received X for them while the question is applicable for the respondents. If the answer is X but the question is not applicable, we consider it as a "*do not* care" item. Table 34 shows all the questions and the correspondent 'X Yes' values. The 'X Yes' values are the Don't Know answers while it is supposed to be known for the users. The higher values in this table are the ones that we call them 'knowledge missing' questions. Any value can be set as *threshold* to identify and highlight the knowledge missing questions. The results demonstrate that the quality point in the knowledge missing items should be emphasized in the TEC Company. A training can be useful to increase the users' knowledge, and this may help the users having a better perception of the SharePoint quality at TEC.

Table 34: The questions with the number of 'X Yes' values categorized by 'Knowledge missing' and 'don't care' labels

Label: Knowledge missing	
Question Title	'X Yes'
1 - Undo availability	8
2 - Help usability	12
3 - Error-message coherence	7
7 - Default values	11
8 - Minimalism	10
9 - Report customization	6
11 - Function key availability	6
13 - Learnability	8
14 - Controllability	9
15 - Secure remote accessibility	10
16 - Error handling agility	9
21 - Controlled modifiability	8
24 - Authentication	6
25 - Authorization	8
26 - Privacy	8
27 - Access rights	7

Label: Do not care	
Question Title	'X Yes'
4 - Clarity in user interfaces	2
5 - Mandatory fields	4
6 - Screen traceability	1
10 - Documentation accuracy	2
12 - Field word processing	5
17 - Responsiveness	3
18 - Satisfaction and interest level	0
19 - Expectancy	3
20 - Import/export options	4
22 - Backward compatibility	5
23 - Inter-operability	2
28 - Concurrency	1

5.7 Correspondance with the reality

Although the participants of the 2nd questionnaire were less technical knowledgeable that the 1st questionnaire participants, we can identify the tech-savvy departments among all TEC departments. In this context, the employees of the tech-savvy departments are those who participated in the 2nd questionnaire and whose daily tasks are more technical and IT oriented than the other department employees. This identification was done by looking at their background educational field and certificates. Table 35 shows the list of tech-savvy departments among the other departments. It shows that the employees of the tech-savvy departments have selected less

Xs than the other employees. To distinguish the tech-savvy departments, we calculated the average number of 'X Yes' per participants, which shows the average of received X while the question is applicable for all participants. For example, in testing department, in average, each participant selected 3.5 X's to the applicable questions.

Table 35: Tech-savvy departments at TEC

Department	***Average number of 'X Yes' per participant***	***Employees Background***	***Label***
Testing	3.5	Engineering	Tech-Savvy
Editing/Translation	5.5	Humanities/Literature	
Electronic data interchange	2.3	Engineering + Business	Tech-Savvy
Executive	1.0	Engineering + Business	Tech-Savvy
Marketing	11.8	Business + Web Design	
Pre-Sales	10.0	Humanities/Business	
Project Development	6.5	Engineering + Business	
R&D	4.0	Engineering	Tech-Savvy
Research Analyst	5.5	Business	
Selection Services	6.5	Business	
Small business groups	6.0	Business	
Vendor Services	4.8	Business	

5.8 Threats to validity

There is a threat that the questions were answered by the employees of the departments who are not involved in SharePoint. For checking the validity and the consistency of our questionnaire, in this section, we show that the departments, which are more involved in SharePoint, are involved in more questions. In other words, we show that the values of $SPIP_D$ and INV_D are positively

correlated. Table 36 shows the values for all the departments. It compares the amount that a given department is involved in TEC's processes, with the amount that the department is involved in the questions. For example, the Testing Department is involved in 65% of TEC's processes in SharePoint while it is involved in 93% of the questions.

Table 36: SharePoint Involvement vs. Questions Involvement

Department	**SPIP (%)**	**INV (%)**
Testing	65	93
Editing/Translation	71	82
Electronic Data Interchange	47	57
Executive	65	82
Marketing	76	86
Pre-Sales	47	68
Project Development	82	89
R&D	88	96
Research Analyst	53	75
Selection Services	41	61
Small Business Groups	59	50
Vendor Services	35	57
Correlation	*0.82*	

The correlation value is *0.82* which is strong enough to support the claim that the departments which are strongly involved in TEC's processes in SharePoint have more related questions in the questionnaire.

5.9 Conclusion

The results demonstrate that the quality point in the knowledge missing items should be emphasized in the TEC Company. A training can be useful to increase the users' knowledge, and this may help the users having a better perception of the SharePoint quality at TEC. The results of this analysis show the field of knowledge missing among the employees. This analysis helps us to find the appropriate training for the employees of each of the department. For example, a general training on the existing UI feature of SharePoint can clarify some of the ambiguities such as undo, or remote access possibility. These training will provide better understanding of the software they are using. Furthermore, the results of this analysis can help to improve the features of SharePoint. The subject of the software improvements can also be extracted from the results of this analysis.

Industry can take advantage of the results of this project. Unfortunately, too often the business leaders neglect the quality aspect of their organizational software systems. They scarify the quality for more functionality, faster development, or lower costs. Our developed artifacts – Expected and Observed Quality – can help the business leaders to have a better vision of current state and future improvements of their software applications. Our expected quality artifact ensures that there are enough activities in software development process to support the user's quality expectations. This will help the software to be more easily accepted by the users. The observed quality artifact can be used to evaluate the user satisfaction of the operational software, and gives a clearer vision for future improvements.

Addressing the Quality Deviation Artifact in practice in form of a case study is recommended as a future work. The quality deviation artifact requires to focus on a software development process from the time that the process gets started until it gets finished. Collecting the data through our questionnaires will lead to generate the quality deviations. Quality deviations will help the business leaders to better plan for next software updates, people training, and business improvements. Unfortunately, during our research period, due to various delays, we did not havec the opportunity to observe a software creation from the start until the end to generate the quality deviations, thus it remains for future researchers to continue this research.

Customizing the questions that compose our questionnaires is also recommended as future work. Since nowadays software is everywhere, addressing the quality of any software with a generic questionnaire may reduce the accuracy, effectiveness, and validity of the survey results. It is

recommended to create the customized and context-oriented questionnaires based on the context of the software under study. For example, the questionnaire addressing the quality of a video game may differ from the questionnaire that aims to evaluate the quality of an embedded software in an industrial control device. In the former, focusing on graphical user interface matters, while in latter, information accuracy and availability is essential.

CHAPTER 6 CONCLUSION AND RECOMMENDATIONS

The main objective of this thesis is to 1) build the Quality Plan Profile by eliciting the expected quality characteristic based on customer quality requirement engineering. 2) Build the Product Quality Profile by quantifying the quality characteristics of the software products. Through this approach, a number of indicator values will be collected measuring the strength of each quality factor, and 3) create the quality deviation artifact (QDA). The QDA is an artifact that shows the deviation between the planned and the observed software quality. Although the QDA was one of our objectives and we theorized the creation of that, unfortunately we could not have a chance to practically create and test it in form of a case study. It remains as a future work for other researcher to follow and improve our methodology in order to create and analyze the QDA.

In this research, we also presented the validation process used to set up the measurement approach that will enable the realization of the QDA diagrams. The quality measurement process in this study is performed using the quality characteristics presented in ISO 25010 and the quality evaluation rules presented in ISO 25021.

A set of questions related to quality measurements factors has been designed, and a team of experts in a software evaluation company was asked to review the questions. According to the results, the experts found 61% of the questions as “Good” while the others should be modified or deleted. Based on the results the questions were reviewed, and the modifications were done accordingly.

The results show that although a general agreement can be seen in the experts’ feedbacks, two experts have the lowest agreement value among the others and the lowest agreement value between each other. This finding shows that an objective evaluation of the quality of a software product can be quite challenging since it highly depends on the evaluator’s perspectives, and such an evaluation must, accordingly considers many stakeholders involvement.

The study reported in this thesis stresses the importance of relying on a quality profile that is likely to outline the various perspectives of the stakeholder's concern by the quality of the product.

In the next stage of the research, we aimed to present a methodology that enables the evaluation of the expected quality characteristics of a software product. For this purpose, we used the collected set of questions related to the quality measurements, which were reviewed iteratively by industry

experts and academic professionals. The final questions formed two working questionnaires, one for end users and one for power users. A software product – in development phase – was selected as a case study and the future users of the software were asked to fill out the questionnaire while taking the software product and their requirements into consideration. Statistical analysis of the responses reveals that end users and power users have the same perspective about the importance of certain quality factors. The results show the feasibility of identifying and quantifying software users' quality-related expectations using the model presented in ISO/IEC 25000. Besides, the results of the case study specifically show that the efficiency and freedom from risk are the most important factors for both types of users in the quality in-use category. In the product quality category, the most important characteristics were security and reliability. Using the Pareto analysis, this research provides the tools for retrieving the causes that affect the importance of any quality characteristic from the users' perspective.

One of the factors that distinguishes high-quality software product from a low-quality one is the degree of "user involvement", which is defined as the level of users' enthusiasm to engage with the software product. Users are generally most willing to engage with software products that not only meet their functional requirements but also their quality expectations. The results of this stage help frame the importance of quality as an essential field of focus during software development and evaluation, and the tools developed – the "quality plan profiles" – enable developers to tailor and specify quality-related requirements in a concrete way.

With quality plan profiles in hand, it is possible to perform a useful crosscheck between users' specific quality expectations and other drivers (functional and architecture/design requirements), before or during the software development process. The crosscheck should be aimed to guarantee that there are enough activities and sub-activities in the software development process to support the users' quality expectations.

In the last stage of the study, the effects of participant's knowledge on the questionnaire results are addressed. The results of our systematic literature review show that the existence of this effect is controversial among the researchers. To examine deeper, we conducted two surveys in a software consulting company. The surveys were targeted to evaluate the quality of Microsoft SharePoint from the end users' point of view at the company. To investigate the participants' knowledge, we focused on the number of 'Don't know' that we received from the participants. The results of our

surveys approve that the participants' knowledge may affect the results of the surveys. According to the obtained results of the two surveys, we can conclude the following points. For simplification, we labeled the 'Don't know' as X in the text.

- *Questions should be applicable:* The questions that have more X, are those questions that are not applicable for most of the users. Thus, the more applicable the questions are, the less X are answered. It means that the participants are more knowledgeable. Participant's relevant knowledge plays a key role to decrease the number of X's.
- *The most beneficial participants should be cherry-picked:* For conducting a survey in software engineering, it is not enough to select appropriate questions or increase the number of participants regardless of their competencies. Our study shows that the participants should be cherry-picked based on their relevant knowledge. The more technical knowledgeable participants we select, the fewer X answers we receive, and consequently, the result of the survey will be more valid.
- *Training is essential:* In a general perspective, the results reveal the idea that the questions that are applicable for most of the users while still many X's have been answered for them, can be considered as training subjects in the organization. The training needed items are those quality elements that are essential, but the users have not relevant knowledge about, so the users need training on those subjects, and that knowledge may enhance the software quality in use.
- *Participants' knowledge of IT is essential:* The departments that have many X's are those that their employees are not tech-savvy. It reveals that increasing the general knowledge of IT of the employees or hiring the tech-savvy employees will affect the perception of the software product quality.

BIBLIOGRAPHY

[1] R. Mirsalari and P. N. Robillard, “Industrial Validation of an Approach to Measure Software Quality,” in *SEDE*, 2015, p. 6.

[2] L. Westfall, *The Certified Software Quality Engineer Handbook*. ASQ Quality Press; Har/Cdr edition, 2009.

[3] I. Standard, “ISO/IEC Systems and Software Engineering—Vocabulary (ISO/IEC/IEEE 24765:2010,” 2015.

[4] Crosby Philip B., *Quality is free*. McGraw-Hill Science/Engineering/Math; 3 edition, 1979.

[5] J. M. Juran, *Juran’s Quality Handbook*, Fifth Edit. New York, USA: McGraw-Hill, 1999.

[6] G. Schulmeyer and J. McManus, *Handbook of Software Quality Assurance*, 3rd Ed. Upper Saddle River, NJ: Prentice Hall PTR, 1998.

[7] L. Arksey, H., & O’Malley, “Scoping studies: towards a methodological framework,” *Int. J. Soc. Res. Methodol.*, vol. 8, no. 1, pp. 19–32, 2005.

[8] H. L. Colquhoun, D. Levac, K. K. O’Brien, S. Straus, A. C. Tricco, L. Perrier, M. Kastner, and D. Moher, “Scoping reviews: Time for clarity in definition, methods, and reporting,” *J. Clin. Epidemiol.*, vol. 67, no. 12, pp. 1291–1294, 2014.

[9] R. Mirsalari and P. N. Robillard, “Expected Software Quality Profile: A methodology and a case study,” in *The 7th IEEE Annual Information Technology, Electronics & Mobile Communication Conference - IEMCON 2016*, 2016, p. 923.

[10] M. Kläs, C. Lampasona, and J. Münch, “Adapting software quality models: Practical challenges, approach, and first empirical results,” in *Proceedings - 37th EUROMICRO Conference on Software Engineering and Advanced Applications, SEAA 2011*, 2011, pp. 341–348.

[11] J. Gottschick and H. Reste, “An empirical evaluation of the quality of interoperability specifications for the web,” in *Proceedings - 36th EUROMICRO Conference on Software Engineering and Advanced Applications, SEAA 2010*, 2010, pp. 398–405.

[12] I. Biscoglio and E. Marchetti, “An experiment of software quality evaluation in the audio-

visual media preservation context," in *Proceedings - 2014 9th International Conference on the Quality of Information and Communications Technology, QUATIC 2014*, 2014, pp. 118–123.

[13] L. Aversano and M. Tortorella, "Analysing the Reliability of Open Source Software Projects," pp. 348–357, 2015.

[14] K. Lochmann, J. Ramadani, and S. Wagner, "Are comprehensive quality models necessary for evaluating software quality?," in *Proceedings of the 9th International Conference on Predictive Models in Software Engineering - PROMISE '13*, 2013, pp. 1–9.

[15] R. Hofman, "Behavioral economics in software quality engineering," *Empir. Softw. Eng.*, vol. 16, no. 2, pp. 278–293, 2011.

[16] L. Corral and I. Fronza, "Better Code for Better Apps: A Study on Source Code Quality and Market Success of Android Applications," in *Proceedings - 2nd ACM International Conference on Mobile Software Engineering and Systems, MOBILESoft 2015*, 2015, pp. 22–32.

[17] P. D. Anjali, "Empirical Validation of Website Quality Using Statistical and Machine Learning Methods," pp. 1–6, 2014.

[18] D. D. J. Suwawi, E. Darwiyanto, and M. Rochmani, "Evaluation of academic website using ISO/IEC 9126," *2015 3rd Int. Conf. Inf. Commun. Technol. ICoICT 2015*, pp. 222–227, 2015.

[19] R. Mirsalari and P. N. Robillard, "Industrial Validation of an Approach to Measure Software Quality," *Sede*, p. 6, 2015.

[20] L. Kumar and S. K. Rath, "Hybrid functional link artificial neural network approach for predicting maintainability of object-oriented software," *J. Syst. Softw.*, vol. 121, pp. 170–190, 2016.

[21] S. Chakrabarty and N. Chaki, "Quality Evaluation of Conceptual Level Object Multidimensional Data Model," *Int. J. Comput. Appl.*, vol. 32, no. 3, p. 14, 2011.

[22] M. Pušnik, M. Heričko, Z. Budimac, and B. Šumak, "XML schema metrics for quality evaluation," *Comput. Sci. Inf. Syst.*, vol. 11, no. 4, pp. 1271–1290, 2014.

[23] Carnegie Mellon University., “Software Engineering Institute (SEI),” 1984. [Online]. Available: www.sei.cmu.edu.

[24] ISO/IEC, “ISO/IEC 25000 - Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE),” 2005.

[25] E. H. Marinho and R. F. Resende, “Quality Factors in Development Best Practices,” in *ICCSA*, 2012, pp. 632–645.

[26] ISO/IEC, “INTERNATIONAL STANDARD ISO/IEC 25010:2010,” 2010.

[27] ISO/IEC, “INTERNATIONAL STANDARD ISO/IEC DIS 25021 — Quality Measure Element,” 2011.

[28] T. Punter, M. Ciolkowski, B. Freimut, I. John, F. Iese, and D.- Kaiserslautern, “Conducting On-line Surveys in Software Engineering Characterizing surveys in SE,” 2003.

[29] T. C. Lethbridge, S. E. Sim, and J. Singer, “Studying Software Engineers : Data Collection Techniques for Software Field Studies,” *Empir. Softw. Eng.*, vol. 10, pp. 311–341, 2005.

[30] B. Kitchenham and S. L. Pfleeger, “Principles of Survey Research Part 5: Populations and Samples,” *ACM SIGSOFT Softw. Eng. Notes*, vol. 27, no. 5, p. 17, 2002.

[31] B. a Kitchenham and S. L. Pfleeger, “Principles of Survey Research Part 3: Constructing a Survey Instrument,” *ACM SIGSOFT Softw. Eng. Notes*, vol. 27, no. 2, p. 20, 2002.

[32] B. Kitchenham and S. L. Pfleeger, “Principles of survey research part 4: questionnaire evaluation,” *ACM SIGSOFT Softw. Eng. Notes*, vol. 27, no. 3, p. 20, 2002.

[33] B. Kitchenham and S. L. Pfleeger, “Principles of survey research part 6: Data Analysis,” *ACM SIGSOFT Softw. Eng. Notes*, vol. 28, no. 2, p. 24, 2003.

[34] B. Kitchenham and S. L. Pfleeger, “Principles of Survey Research,” *ACM SIGSOFT Softw. Eng. Notes*, vol. 26, no. 6, p. 16, 2001.

[35] B. a Kitchenham and S. L. Pfleeger, “Principles of Survey Research Part 2 : Designing a Survey Sample size Experimental designs,” *Softw. Eng. Notes*, vol. 27, no. 1, pp. 18–20, 2002.

[36] H. Al-Kilidar, K. Cox, and B. Kitchenham, “The use and usefulness of the ISO/IEC 9126

quality standard," in *International Symposium on Empirical Software Engineering.*, 2005, pp. 122–128.

[37] R. E. Al-qutaish, "A Maturity Model of Software Product Quality," *Res. Pract. Inf. Technol.*, vol. 43, no. 4, pp. 307–328, 2010.

[38] E. Van Veenendaal, R. Hendriks, and R. Van Vonderen, "Measuring Software Product Quality," *SQP*, vol. 5, no. 1, pp. 6–13, 2002.

[39] C. Wohlin, P. Runeson, M. Host, Magnus C. Ohlsson, Bjorn Regnell, and Anders Wesslen, *Experimentation in Software Engineering*. 2012.

[40] P. Torino and P. Torino, "A State-of-the-Practice Survey of Risk Management in Development with Off-the-Shelf A State-of-the-Practice Survey on Risk Management in Development with Off- The-Shelf Software Components," vol. 34, no. April, pp. 271–286, 2016.

[41] A. Nugroho and M. R. V. Chaudron, "A survey into the rigor of UML use and its perceived impact on quality and productivity," *Proc. Second ACM-IEEE Int. Symp. Empir. Softw. Eng. Meas. - ESEM '08*, p. 90, 2008.

[42] J. Soini, "A Survey of Metrics Use in Finnish Software Companies," *2011 Int. Symp. Empir. Softw. Eng. Meas.*, pp. 49–57, 2011.

[43] W. Chen, J. Li, J. Ma, R. Conradi, J. Ji, and C. Liu, "A survey of software development with open source components in Chinese software industry," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 4470 LNCS, pp. 208–220, 2007.

[44] J. Li, J. Ma, R. Conradi, W. Chen, J. Ji, and C. Liu, "A survey on the business relationship between Chinese outsourcing software suppliers and their outsourcers," *Proc. - Asia-Pacific Softw. Eng. Conf. APSEC*, pp. 470–477, 2007.

[45] E. Petrinja, A. Sillitti, and G. Succi, "Adoption of OSS Development Practices by the Software Industry: A Survey," *Open Source Syst. Grounding Res.*, vol. 365, pp. 233–243, 2011.

[46] A. Macphail, T. Hainey, and T. M. Connolly, "Applying Mlearning in Software Engineering

Education : a Survey of Mobile Usage," *Mob. Learn.*, no. iii, 2012.

[47] M. Usman, E. Mendes, and J. Börstler, "Effort estimation in Agile software development: A survey on the state of the practice," *ACM Int. Conf. Proceeding Ser.*, vol. 27–29–Apri, 2015.

[48] R. N. Memon, S. S. Salim, and R. Ahmad, "Identifying research gaps in requirements engineering education: An analysis of a conceptual model and survey results," *2012 IEEE Conf. Open Syst.*, pp. 1–6, 2012.

[49] Y. Cerqueira, S. R. De Lemos, Y. C. . Cavalcanti, P. A. . Da Mota Silveira Neto, I. . Do Carmo Machado, E. S. . De Almeida, and S. R. . De Lemos Meira, "Towards Understanding Software Change Request Assignment : a survey with practitioners," *ACM Int. Conf. Proceeding Ser.*, pp. 195–206, 2013.

[50] C. Palomares, C. Quer, and X. Franch, "Requirements reuse and requirement patterns : a state of the practice survey," *Empir. Softw. Eng.*, 2016.

[51] David Ameller, M. Galster, P. Avgeriou, and X. Franch, "A survey on quality attributes in service-based systems," *Softw. Qual. J.*, vol. 24, no. 2, pp. 337–364, 2016.

[52] H. Tsuji, A. Sakurai, K. Yoshida, A. Tiwana, and A. Bush, "Questionnaire-Based Risk Assessment Scheme for," pp. 114–127.

[53] H.-C. Huang, "Freemium business model: construct development and measurement validation," *Internet Res.*, vol. 26, no. 3, pp. 604–625, 2016.

[54] D. Tofan, M. Galster, P. Avgeriou, and D. Weyns, "Software engineering researchers' attitudes on case studies and experiments: An exploratory survey," *Eval. Assess. Softw. Eng. (EASE 2011), 15th Annu. Conf.*, no. 638, pp. 91–95, 2011.

[55] O. Albayrak, "Instructor's Acceptance of Games Utilization in Undergraduate Software Engineering Education: A Pilot Study in Turkey," *2015 IEEE/ACM 4th Int. Work. Games Softw. Eng.*, pp. 43–49, 2015.

[56] D. Budgen, B. A. Kitchenham, S. M. Charters, M. Turner, P. Brereton, and S. G. Linkman, "Presenting software engineering results using structured abstracts: A randomised experiment," *Empir. Softw. Eng.*, vol. 13, no. 4, pp. 435–468, 2008.

[57] F. Q. B. Da Silva and A. C. C. Frana, “Towards understanding the underlying structure of motivational factors for software engineers to guide the definition of motivational programs,” *J. Syst. Softw.*, vol. 85, no. 2, pp. 216–226, 2012.

[58] I. Erfurth and W. R. Rossak, “A look at typical difficulties in practical software development from the developer perspective A field study and a first solution proposal with UPEX,” *Proc. Int. Symp. Work. Eng. Comput. Based Syst.*, pp. 241–248, 2007.

[59] N. Fenton, M. Neil, W. Marsh, P. Hearty, Ł. Radliński, and P. Krause, “On the effectiveness of early life cycle defect prediction with Bayesian nets,” *Empir. Softw. Eng.*, vol. 13, no. 5, pp. 499–537, 2008.

[60] I. Garcia, C. Pacheco, and P. Sumano, “Use of questionnaire-based appraisal to improve the software acquisition process in small and medium enterprises,” *Stud. Comput. Intell.*, vol. 150, pp. 15–27, 2008.

[61] J. Hutchinson, J. Whittle, M. Rouncefield, and S. Kristoffersen, “Empirical assessment of MDE in industry,” *2011 33rd Int. Conf. Softw. Eng.*, pp. 471–480, 2011.

[62] M. V. Kosti, R. Feldt, and L. Angelis, “Personality, emotional intelligence and work preferences in software engineering: An empirical study,” *Inf. Softw. Technol.*, vol. 56, no. 8, pp. 973–990, 2014.

[63] T. Sedano, “Code readability testing, an empirical study,” *Proc. - 2016 IEEE 29th Conf. Softw. Eng. Educ. Training, CSEEandT 2016*, pp. 111–117, 2016.

[64] P. . Diebold, A. . Vetró, and D. . Méndez Fernández, “An Exploratory Study on Technology Transfer in Software Engineering,” *Int. Symp. Empir. Softw. Eng. Meas.*, vol. 2015–Novem, pp. 86–95, 2015.

[65] A. Forward and T. C. Lethbridge, “Problems and Opportunities for Model-Centric Versus Code-Centric Software Development: A Survey of Software Professionals,” *Proc. 2008 Int. Work. onModels Softw. Eng.*, pp. 27–32, 2008.

[66] J. Ji, J. Li, and R. Conradi, “Some lessons learned in conducting software engineering surveys in China,” *ESEM’08 Proc. 2008 ACM-IEEE Int. Symp. Empir. Softw. Eng. Meas.*, pp. 168–177, 2008.

[67] J. M. Rojas, G. Fraser, and A. Arcuri, “Automated Unit Test Generation During Software Development: A Controlled Experiment and Think-aloud Observations,” pp. 338–349, 2015.

[68] P. Karpati, Y. Redda, A. L. Opdahl, and G. Sindre, “Comparing attack trees and misuse cases in an industrial setting,” *Inf. Softw. Technol.*, vol. 56, no. 3, pp. 294–308, 2014.

[69] J. P. Campos, J. L. Braga, A. M. de Resende, and C. H. Os’orio Silva, “Identification of Aspect Candidates by Inspecting Use Cases Descriptions,” *SIGSOFT Softw. Eng. Notes*, vol. 35, no. 4, pp. 1–9, 2010.

[70] L. Prechelt and M. Liesenberg, “Design patterns in software maintenance: An experiment replication at Freie Universit??t Berlin,” *Proc. - 2011 2nd Int. Work. Replication Empir. Softw. Eng. Res. RESER 2011*, pp. 1–6, 2012.

[71] M. Haddara and A. Elragal, “ERP adoption cost factors identification and classification: a study in SMEs,” *Int. J. Inf. Syst. Proj. Manag.*, vol. 1, no. 2, pp. 5–21, 2013.

[72] A. Jedlitschka, “Evaluating a model of software managers’ information needs,” *Proc. 2010 ACM-IEEE Int. Symp. Empir. Softw. Eng. Meas. - ESEM ’10*, p. 1, 2010.

[73] M. Sensalire, P. Ogao, and A. Telea, “Evaluation of software visualization tools: Lessons learned,” *2009 5th IEEE Int. Work. Vis. Softw. Underst. Anal.*, pp. 19–26, 2009.

[74] M. Schmidberger and B. Brugge, “Need of Software Engineering Methods for High Performance Computing Applications,” *Parallel Distrib. Comput. (ISPDC), 2012 11th Int. Symp.*, pp. 40–46, 2012.

[75] D. Galin, *Software Quality Assurance From theory to implementation*. 2004.

[76] S. H. Kan, *Metrics and Models in Software Quality Engineering*. Addison-Wesley Longman Publishing Co., Inc., 2002.

[77] Rohaiza Abd. Rokis, “Youth Employability and Work Attitudes,” *Int. J. Sci. Commer. Humanit.*, vol. 2, no. 5, p. 13, 2014.

[78] G. Vigderhous, *The Level of Measurement and Permissible Statistical Analysis in Social Research*, vol. 20, no. 1. 1977.

[79] U. Jakobsson, “Statistical presentation and analysis of ordinal data in nursing research,”

Scand. J. Caring Sci., vol. 18, no. 4, pp. 437–440, 2004.

[80] IEEE Computer Society, "IEEE Standard for a Software Quality Metrics Methodology - IEEE Std 1061™-1998 (R2009)," vol. 1998, 2009.

[81] H. Jung and S. Kim, "Measuring Software Product Quality: A Survey of ISO/IEC 9126," *IEEE Softw.*, pp. 88–92, 2004.

[82] L. M. Lozano, E. García-Cueto, and J. Muñiz, "Effect of the Number of Response Categories on the Reliability and Validity of Rating Scales," *Methodol. Eur. J. Res. Methods Behav. Soc. Sci.*, vol. 4, no. 2, pp. 73–79, Jan. 2008.

[83] C. Alzola and F. Harrell, "An Introduction to S and the Hmisc and Design Libraries," 2006. [Online]. Available: http://her.gr.distfiles.macports.org/mirrors/CRAN/doc/contrib/Alzola+Harrell-Hmisc-Design-Intro.pdf. [Accessed: 29-Jan-2015].

[84] "Technology Evaluation Centers." [Online]. Available: http://www.technologyevaluation.com/. [Accessed: 18-Apr-2017].

[85] J. Baroudi, M. Olson, and B. Ives, "An empirical study of the impact of user involvement on system usage and information satisfaction," *Commun. ACM*, vol. 29, no. 3, pp. 232–238, 1986.

[86] Software Engineering Institute, "CMMI® for Development, Version 1.3," 2010.

[87] IEEE Computer Society, *SWEBOK Guide*. 2014.

[88] S. Hansen and J. Rennecker, "Getting on the same page: Collective hermeneutics in a systems development team," *Inf. Organ.*, vol. 20, no. 1, pp. 44–63, Jan. 2010.

[89] O. Braud, "Facteurs decisionnels pour l'iplantation d'un ERP dans les PME : Le role de l'evaluation des benefices tangibles et intangibles," 2008.

[90] A. J. Albrecht, "Measuring application development productivity," *IBO Conf. Appl. Dev.*, vol. 10, pp. 83–92, 1979.

[91] R. Alvaro, "Framework for a global quality evaluation of a website," *Online Inf. Rev.*, vol. 36, no. 3, pp. 347–382, 2012.

[92] H. Yang, “Measuring software product quality with ISO standards base on fuzzy logic technique,” *Adv. Intell. Soft Comput.*, vol. 137 AISC, pp. 59–67, 2012.

[93] P. Tomas, M. J. Escalona, and M. Mejias, “Open source tools for measuring the Internal Quality of Java software products. A survey,” *Comput. Stand. Interfaces*, vol. 36, no. 1, pp. 244–255, 2013.

[94] M. Kwiatkowski and C. Verhoef, “Recovering management information from source code,” *Sci. Comput. Program.*, vol. 78, no. 9, pp. 1368–1406, 2013.

[95] S. Lehtonen, “Metrics for Gerrit code reviews,” no. May, pp. 31–45, 2015.

[96] B. H. Layne, J. R. Decristoforo, and D. Mcginty, “Electronic versus traditional student ratings of instruction,” *Res. High. Educ.*, vol. 40, no. 2, pp. 221–232, 1999.

[97] B. Kitchenham, O. Pearl Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, “Systematic literature reviews in software engineering - A systematic literature review,” *Inf. Softw. Technol.*, vol. 51, no. 1, pp. 7–15, 2009.

[98] Ameller David, Galster Matthias, Paris Avgeriou, and Franch Xavier, “A survey on quality attributes in service-based systems,” *Softw. Qual J*, 2016.

[99] K. Dullemond and B. van Gameren, “What Distributed Software Teams Need to Know and When: An Empirical Study,” *Glob. Softw. Eng. (ICGSE), 2013 IEEE 8th Int. Conf.*, pp. 61–70, 2013.

[100] J. M. Rojas, G. Fraser, and A. Arcuri, “Automated unit test generation during software development: a controlled experiment and think-aloud observations,” *Proc. 2015 Int. Symp. Softw. Test. Anal. - ISSTA 2015*, pp. 338–349, 2015.

[101] A. Jedlitschka, “Evaluating a model of software managers’ information needs,” *Proc. 2010 ACM-IEEE Int. Symp. Empir. Softw. Eng. Meas. - ESEM ’10*, p. 1, 2010.

[102] A. Nugroho and C. F. J. Lange, “On the relation between class-count and modeling effort,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 5002 LNCS, pp. 93–104, 2008.

APPENDICES

APPENDIX A – ARTICLE 1: INDUSTRIAL VALIDATION OF AN APPROACH TO MEASURE SOFTWARE QUALITY

Reza Mirsalari
Laboratoire de génie logiciel
École Polytechnique de Montréal
Montréal, Canada
reza.mirsalari@polymtl.ca

Mehdi Aftahi
Research and Development Technology Evaluation Centers
Montréal, Canada
mehdi@tec-centers.com

Pierre N. Robillard
Laboratoire de génie logiciel
École Polytechnique de Montréal
Montréal, Canada
pierre-n.robillard@polymtl.ca

Accepted at SEDE 2015 conference

Abstract

Software quality has emerged into a multi-faceted discipline that requires the software product to satisfy a wide range of stakeholders. Comprehensive specifications and evaluation of the quality of software are essential factors in ensuring value to stakeholders. Ensuring that a software product will add value to the stakeholders requires the software quality to be measured at specified milestones of the development process. The deviation between the expected and the measured quality attributes of the final product shows how much of the stakeholders' requirements are satisfied. The main objective of this study is moving towards providing the quality deviation artifact. The rules and quality characteristics presented in ISO/IEC 25000 series of standard are applied to define a series of quality-related questions. The questions have been reviewed by the software evaluation experts in a company. The results show the importance of relying on a quality profile that is likely to outline the various perspectives of the stakeholders concern by the quality of the product.

1 Introduction

Software quality is defined as the comprehensive set of characteristics that enable the product to satisfy the stakeholders' needs. Accordingly, "software quality is fundamental to software success" [36]. Furthermore, evaluation of software quality is essential, since inadequate quality in a software product may lead to human or financial losses [37] while high-quality are fundamental to providing value, and avoiding potential negative effects for the stakeholders.

Taking a broader perspective, high-quality software is recognized as a product that has been specified correctly, and that meets its expected specifications. Software products meeting the stakeholder's requirements are more likely to be accepted and utilized by the stakeholders [2]. Comprehensive specification and evaluation of the quality of software is an essential factor in ensuring value to stakeholders. This added value can be achieved by defining the desired quality characteristics associated with the stakeholders' goals and objectives. These features help to represent the quality of the software products from the perspective of that particular characteristic.

It is important that the quality characteristics are specified, measured and evaluated. It is shown that the international standard ISO/IEC 25000 [24] is capable of assessing the quality of a broad range of software applications; from traditional to new application classes such as smart mobile devices [25]. The quality model in ISO/IEC 25010 is designed to identify relevant quality characteristics for software products, which can be used to establish requirements, criteria for satisfaction and the corresponding measures [26]. ISO/IEC 25010 *d*escribes the quality model encompassing the characteristics and sub-characteristics for software quality in use, and software product quality.

In addition to the ISO/IEC 25010 that provides the quality characteristics, an approach is also needed to specify the quality measurements. In this work, we applied the ISO/IEC 25021 [27] which provides a set of rules to design and verify the appropriate quality measure elements.

The main objective in this study is to create the software quality deviation artifact through comparing the user expected quality against the final observed quality of a software product. The steps for creating the software quality deviation artifact are presented in section 2. To make this comparison feasible, it is intended to apply the ISO/IEC 25000 standard, as a tool to measure the quality of the two above mentioned artifacts. Section 3 presents the quality measurement process. At the final stage, the quality deviation artifact will show the comparison results of the planned versus observed quality characteristics. The preliminary results of an industrial experiment performed in a software evaluation company are discussed in section 4. The section 5 presents the conclusion.

2 Software quality deviation artifact

Many publications in the literature demonstrate the various aspects of software quality assessment. From the 1970s up to date, several software quality models have been proposed such as McCall, Boehm, Dromey, FURPS, ISO 9126 and ISO 25000 (SQuaRE).

The software quality is evaluated by a collection of relevant quality characteristics which are measured by applying a measurement method. A measurement method is a logical classification of operations applied to quantify attributes with respect to a specified scale. During this process, software quality measures turn into quantifications of the quality characteristics. Not each and every quality characteristics are of equal importance to a software product. A method will be used to identify the most important quality characteristics by means of a risk assessment, to establish achievement criteria, and to finally measure the quality characteristics using the ISO/IEC 25010 standard. The quality characteristics can be addressed at the beginning of a development process to discover the expected software quality, and at the end of the development process, that leads to software product quality. This can be achieved by interviewing stakeholders inside the project (such as the developers, product manager, the project manager, configuration manager, etc.) and outside the project (the various types of users are important) [38].

In the current research project, the objective is to 1) build the Quality Plan Profile by eliciting the expected quality characteristic based on customer quality requirement engineering. 2) Build the Product Quality Profile by quantifying the quality characteristics of the software products. Through this approach, a number of indicator values will be collected measuring the strength of each quality factor, and 3) create the quality deviation artifact (QDA). The QDA is a report that shows the deviation between the planned and the observed software quality.

The stepwise process illustrated in Figure 15 represents the steps and the roles in our quality engineering process. The process starts with eliciting the expected customer's quality by the collaboration of quality engineering team and the customer. The next step is to quantify the elicited quality factors that is performed by applying the ISO/IEC 25000 approach. The ISO/IEC 25000 relates the preferences and requirements of stakeholders regarding the software product to the standard software quality characteristics. The stakeholders' needs and requirements are investigated and identified using a structured questionnaire. The questionnaire consists of a number of questions regarding product and process quality characteristics. These characteristics are intended to be expressed in the stakeholders' language. Instead of asking whether "usability" is important, one asks questions about the qualifications of the users that impact the usability requirements, for example, the quantity of users, their involvement with the product (or a similar one), and their educational level. Similar questions are asked about the other quality characteristics. The answers given are used to identify the most relevant quality characteristics and the related values. The detailed steps of the quality measurement process are explained in the section III.

The aim of step 3 is to build the quality plan profile. The quality engineering team provides the quality plan profile that is a diagram showing the quantified values of elicited quality requirements. The radar chart illustrated in Figure 16 represents the planned quality characteristics which are expected by the customer.

The diagrams in Figure 16 illustrate an example of the measured quality characteristics of a hypothetical software product. These diagrams display a quality figure for the software products. As such, they are called Quality Profile Diagrams (QPD). The scale on the radar Axis is from 0 to 100 for illustrative purpose only. The appropriate scale will be determined during building of the quality model. For example in some cases, it may turn out that an ordinal scale may be more appropriate, with qualifiers like "excellent", "satisfactory", "to be improved", "unsatisfactory" and "inappropriate".

The QPD's in Figure 16 shows that according to the customer quality requirements, the characteristics such as Portability, Functional suitability and Reliability are the most important factors that should be taking into consideration during the development and QA phases according to the planned quality profile. What is actually observed is that the product has a major weakness in portability and in-turn it has a high level of compatibility. Although it is an advantage for a software product to be compatible with other working environments, however, it cannot be validated since it is not according to the customers' requirements. Therefore, the client pays for a quality factor which is not needed.

In step 4 in Figure 15 the actual development process takes place. The software developers form the team; the development method is selected, and project management considerations are taken into account. Concurrently the software quality assurance activities are performed by the quality engineering team.

The quality assurance includes the preventive activities that help the development process to come up with a product as flawless as possible. The result of development and quality assurance process is the executable software product that can be deployed on the customer's site, which is shown as step 5 in Figure 15.

After the software product is delivered to the customer, the next step is to measure the product quality by performing the quality control actions, which is step 8 in Figure 15. The goal is to evaluate the expected quality elements that had been highlighted during the quality planning phases; i.e. step 1 through step 3 in Figure 15.

The product quality evaluation is performed using a questionnaire and a quality model in order to create the product quality profile, which is step 7 in Figure 15. The method for creating this profile is similar to the method used to create the quality plan profile.

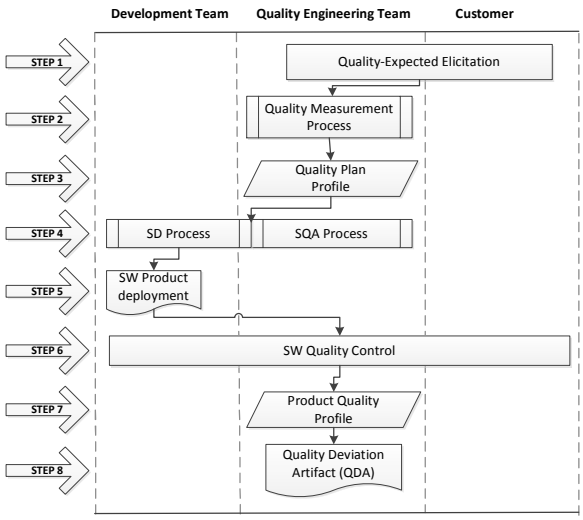


Figure 21: Method to build quality deviation artifact

The final step, step 8 in Figure 15, is aimed to determine the deviation of the expected quality values which had been identified before development phase, and the actual quality values that are specified after the product quality is evaluated.

3 Quality Measurement Process

3.1 Quality Model

As explained above, creating the quality profile needs to apply the quality measure elements in a quality model. The elements and the sequential steps of the quality measurement model are illustrated in Figure 17. As it can be seen in this model, quality attributes, known as "Quality Measure Elements (QME)", are associated to a question. When a data owner answers a question, the value of a QME is specified. Designing the answer scales should be done with care. If there are too few rating-scale categories, the answer may not capture the questions' full

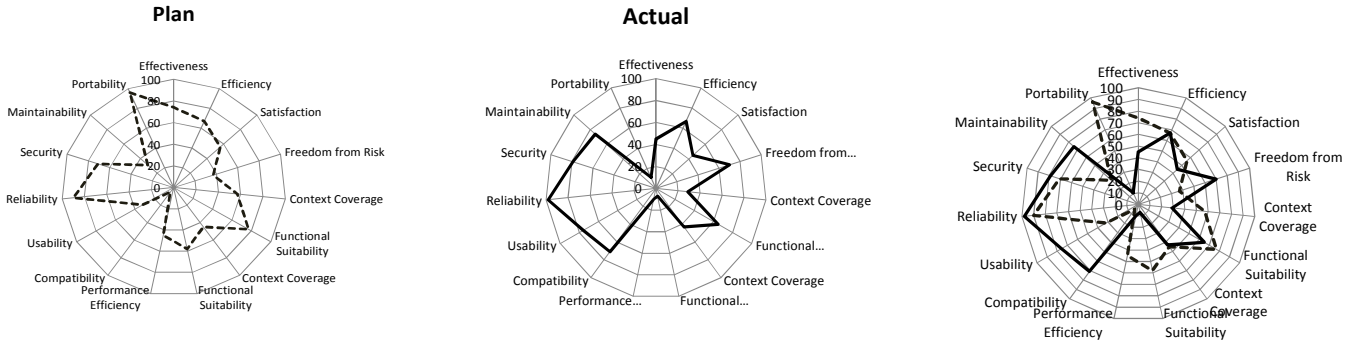


Figure 22: Software Quality Profile diagrams (QPD) for the planned quality and the observed quality (Actual) and quality deviation diagram between the two profile diagrams.

discriminative power [80]. On the other hand, if there are too many categories they might be beyond the respondents' limited discriminatory powers. "*A Monte Carlo study of the number of scale response categories' effects on reliability and validity of rating scales showed that as the number of response alternatives increases, both reliability and validity improve. The optimum number of alternatives is between four and seven. With fewer than four alternatives the reliability and validity decrease, and from seven alternatives onwards psychometric properties of the scale scarcely increase further*" [81][82]. Therefore, based on the context of the questions, we adopted ten different rating scales. The rating scales are presented in Table 28. All rating scales include a "*not related*" and "*don't know*" option. Using this scale, we measured the respondents' evaluations. For example, one question that asked "How do you evaluate the software documentation correctness?" is associated to "Poor - Excellent" rating scale. Those ratings are then converted into numerical. It is usual for a questionnaire to have missing values as well as "don't know" responses. Our study considered "don't know" as a missing value [81]. A statistical technique called *imputation* is used to estimate the missing values [83]. Therefore, all answers are important, even missing ones, for the data analysis.

Table 37 : Rating scales

Absolute Yes/No	**%100 - %0 of his/her time**	**1h - 1w**	**High-Low Probability**	**All-None**	**Agree-Disagree**	**Yes/No**	**Quality Numeric**	**Poor - Excellent**	**Support**
Absolutely No	%75 - %100 of his/her time	Up to 1 hour	High (%70-%100)	None	Disagree	Yes	Number 1-10000	Poor	Unsupported
Yes - for %20 to %50 of cases	%50 - %75 of his/her time	1 hour to 24 hours	Medium (%30-%70)	Least	Mostly agree	No		Fair	Poorly supported
Yes - for %50 to %80 of cases	%25 - %50 of his/her time	1 day to 1 week	Low (%0-%30)	Most	completely agree			Good	Fairly supported
Absolutely Yes	%0 - %25 of his/her time	More than 1 week		All				Very Good	Partially supported
								Excellent	Mostly supported
									Fully supported

The QME's are measured by applying a measurement function. A measurement function (or measurement method) is the "*algorithm or calculation performed to combine two or more quality measure elements. A Measurement function is applied to the QME to generate a Quality Measure (QM)*" [27].

Measurement functions use QME's to determine quality measures (QM). Quality Measure (QM) is defined as a "*derived measure that is defined as a measurement function of two or more values of quality measure elements*" [27].

The quality characteristics (QCh) and quality sub-characteristics (QSub) can be quantified after the quality measure determination process. Quality characteristic (QCh) is defined as "*the inherent property of an entity that can be distinguished quantitatively or qualitatively by human or automated means. Quality characteristic is the category of software quality attributes that bears on software quality. The software quality characteristics may be refined into multiple levels of sub-characteristics and finally into software quality attributes*" [27].

3.2 Weighting of quality measures

Our quality measurement model requires a weighting ratio to be assigned to each QM when they are associated with quality sub-characteristics. Theoretically, the weights are the values that determine the importance of the child that composes the parent component (per Figure 17).

Not all the children have the same importance from the parent point of view. For each child, the weight is a coefficient that determines the importance of the child value with respect to the others. For example, as shown in Table 29, "flexibility" is a quality sub-characteristic (QSub) that is decomposed into three quality measures (QM) with a specific weight for each QM.

In our measurement method, the weight ω is a decimal between 0 and 1. The 0 value indicates that this factor should not be taken into consideration while the 1 value indicates that this factor is the only one to be considered. The total weight values for each QSub are normalized to one.

The process used to specify the weight for quality sub-characteristics is the same as the one between QM and QSub. The quality measurement model also uses a weighting ratio to assign the QSub to the QCh. For example "compatibility" is a QCh that can be decomposed to two QSub's, with a specific weight for each QSub.

Likewise, quality characteristics have different weights for different software products in different domains. For example, for an organization requiring a software product for three-month municipal property tax calculation, the "maintainability" should not be that important; while the "reliability" might be essential.

Table 38: An example of weight determination

Quality sub characteristic $(QSub)$	**Quality Measure** (QM)	**Weight (importance)** (ω)
Flexibility	Flexible context of use	0.1
	Flexible design features	0.4
	Software Flexibility	0.5
	Total	***1***

Finally, the total quality value of a given software product β is the total value of each weighted quality characteristic values defined as the 13 quality characteristics in the ISO/IEC 25010.

Therefore, we have,

$$Quality_{\beta} = \sum_{k=1}^{13} \omega_k . QCh_k$$

Where,

$$\sum_{k=1}^{13} \omega_k = 1$$

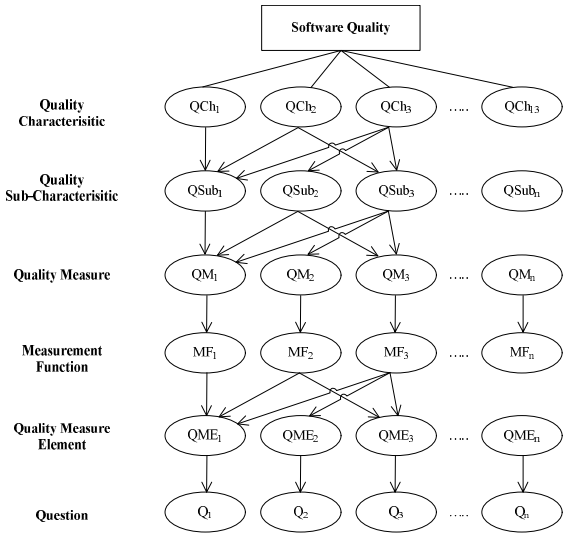


Figure 23: Quality measurement model based on ISO 25021

4 Results and Discussions

The data for this study were obtained from the Technology Evaluation Centers (TEC) knowledgebase. "*TEC helps other companies to investigate, evaluate, and select the best enterprise software solutions for their unique business requirements. From small businesses to large enterprises, its clients include hundreds of private- and public-sector organizations in a variety of industries. A complete list of vendors and products serves the TEC software selection methodology supporting the software acquirers to make an efficient decision*" [84].

As described earlier, the first step in creating a quality profile is requirement elicitation. In the current study, the researcher designed 151 questions related to quality measure elements. In addition to the questions, 10 different rating scales are also designed and are assigned to each question. The list of rating scales is presented in Table 28.

We identified the data owners, who were the most likely appropriate for answering the various questions. A team of 7 experts working in TEC were asked to review the questions and determine if the questions and their rating scales are appropriate and relevant. The experts, who were selected by the company's CTO, are staff members with more than 10 years of experience in the field of software acquiring, enterprise implementation and deployment. The experts were also required to validate the questions by providing one of the 9 feedback types on the relevance of each of the questions. The feedback types which are listed in Table 8. The experts also could write a free text as comments for each question. The numbers of received comments are presented in Table 7. The results presented in Figure 18 shows that in average, 61% of the respondents found the questions and the rating scales as "Good".

Table 39: Number of comments per expert

E1	E2	E3	E4	E5	E6	E7
45	25	42	56	35	111	49

This experiment was coordinated by the researcher. The experts, in isolation from one another, used their personal experience and judgment to evaluate the questions and send their feedback. The coordinator collected the feedbacks and prepared a summary of expert's feedbacks, which is called "composite report". At the end, each expert receives the composite report and was asked to make new evaluation based

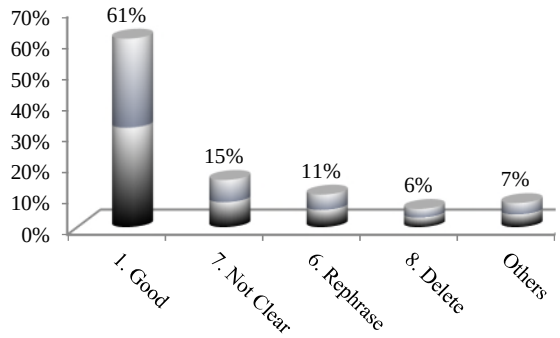


Figure 24: The average percentage of each received feedback type

on the feedback of others. This experiment is based on Delphi method that is detailed in [2].

The obtained data can also be viewed from another perspective. The data presented in Table 9 shows the number of answers that are identical between the experts. For example, 85 options have been identically selected by the experts E3 and E5. The table shows that the expert E2 has the best agreement while the experts E6 and E7 have the least agreement value among the other experts.

Table 40 : Feedback Types

Option	**Feedback description**
1. Good	The question and its rating scale are appropriate
2. Change Rating Scale	The associated rating scale doesn't fit for this question
3. Split the question	The question can be decomposed into 2 questions as…
4. Combine	This question can be combined with the question number X
5. Duplicate->Remove	The question is duplicated with question number X
6. Rephrase	The phrasing is not appropriate. I suggest this "…"
7. Not Clear	It is not clear that what the question will measure
8. Delete	The question is irrelevant. It can't be answered.
9. Don't know	Out of my expertise

In addition to that, it can be seen that the agreement value between the experts E6 and E7 is the lowest value in the table. Although these two experts have almost the same working experiment and academic education level, the results show that their viewpoints are different. Each question was revised based on the received feedbacks and comments. Since the "Not Clear" and "Rephrase" were the most selected options after "Good", 63 questions were rephrased, and 40 rating scales were changed. Figure 6 shows the number of modifications that were made for the whole questionnaire. Some questions had two modifications; consequently, the summation of the numbers in Figure 6 does not add up to 151.

Table 41: Number of identical feedbacks

	E1	E2	E3	E4	E5	E6	E7
E1		**109**	90	78	88	51	49
E2			99	91	91	50	51
E3				75	85	53	54
E4					80	52	52
E5						57	55
E6							**48**
E7							

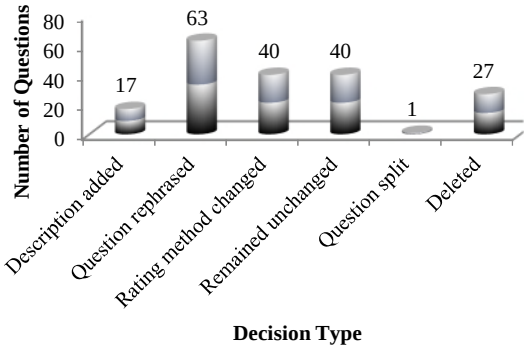


Figure 25: Number of questions for each decision

5 Conclusion

The main objective of this research project is to provide the software quality deviation artifact (QDA). We defined the QDA as an artifact the represents the deviation between expected quality and observed quality of a software product. Providing the QDA needs the software quality to be measured at the beginning of the project on the basis of the customer quality expectations, and the end of the development process on the basis of the produced software.

The purpose of this paper is to present the validation process used to set up the measurement approach that will enable the realization of the QDA diagrams. The quality measurement process in this study is performed using the quality characteristics presented in ISO 25010 and the quality evaluation rules presented in ISO 25021.

A set of questions that are related to quality measure elements has been designed, and a team of experts in a software evaluation company were asked to review the questions. According to the results, the experts found 61% of the questions as "Good" while the others should be changed or deleted. Based on the results the questions were reviewed, and the modifications were done accordingly.

The results show that although a general agreement can be seen in the experts' feedbacks, two experts have the lowest agreement value among the others and the lowest agreement value between each other. This finding shows that an objective evaluation of the quality of a software product can be quite challenging since it depends very much on the evaluator's perspectives, and such an evaluation must, accordingly takes into account the many stakeholders involved.

The study reported in this paper stresses the importance of relying on a quality profile that is likely to outline the various perspectives of the stakeholders concern by the quality of the product.

In the future, it is planned to associate the reviewed quality measures with the standard quality characteristics to provide the QDA through a case study.

Acknowledgment

Many thanks to all participants at Technology Evaluation Centers (TEC), for their continued support and helpful input.

This work was funded in part by the Canadian Mitacs-Accelerate program, grant number IT04215.

References

[1] R. Mirsalari and P. N. Robillard, "Industrial Validation of an Approach to Measure Software Quality," in *SEDE*, 2015, p. 6.

[2] L. Westfall, *The Certified Software Quality Engineer Handbook*. ASQ Quality Press; Har/Cdr edition, 2009.

[3] I. Standard, "ISO/IEC Systems and Software Engineering—Vocabulary (ISO/IEC/IEEE 24765:2010," 2015.

[4] Crosby Philip B., *Quality is free*. McGraw-Hill Science/Engineering/Math; 3 edition, 1979.

[5] J. M. Juran, *Juran's Quality Handbook*, Fifth Edit. New York, USA: McGraw-Hill, 1999.

[6] G. Schulmeyer and J. McManus, *Handbook of Software Quality Assurance*, 3rd Ed. Upper Saddle River, NJ: Prentice Hall PTR, 1998.

[7] L. Arksey, H., & O'Malley, "Scoping studies: towards a methodological framework," *Int. J. Soc. Res. Methodol.*, vol. 8, no. 1, pp. 19–32, 2005.

[8] H. L. Colquhoun, D. Levac, K. K. O'Brien, S. Straus, A. C. Tricco, L. Perrier, M. Kastner, and D. Moher, "Scoping reviews: Time for clarity in definition, methods, and reporting," *J. Clin. Epidemiol.*, vol. 67, no. 12, pp. 1291–1294, 2014.

[9] R. Mirsalari and P. N. Robillard, "Expected Software Quality Profile: A methodology and a case study," in *The 7th IEEE Annual Information Technology,*

Electronics & Mobile Communication Conference - IEMCON 2016, 2016, p. 923.

[10] M. Kläs, C. Lampasona, and J. Münch, "Adapting software quality models: Practical challenges, approach, and first empirical results," in *Proceedings - 37th EUROMICRO Conference on Software Engineering and Advanced Applications, SEAA 2011*, 2011, pp. 341–348.

[11] J. Gottschick and H. Reste, "An empirical evaluation of the quality of interoperability specifications for the web," in *Proceedings - 36th EUROMICRO Conference on Software Engineering and Advanced Applications, SEAA 2010*, 2010, pp. 398–405.

[12] I. Biscoglio and E. Marchetti, "An experiment of software quality evaluation in the audio-visual media preservation context," in *Proceedings - 2014 9th International Conference on the Quality of Information and Communications Technology, QUATIC 2014*, 2014, pp. 118–123.

[13] L. Aversano and M. Tortorella, "Analysing the Reliability of Open Source Software Projects," pp. 348–357, 2015.

[14] K. Lochmann, J. Ramadani, and S. Wagner, "Are comprehensive quality models necessary for evaluating software quality?," in *Proceedings of the 9th International Conference on Predictive Models in Software Engineering - PROMISE '13*, 2013, pp. 1–9.

[15] R. Hofman, "Behavioral economics in software quality engineering," *Empir. Softw. Eng.*, vol. 16, no. 2, pp. 278–293, 2011.

[16] L. Corral and I. Fronza, "Better Code for Better Apps: A Study on Source Code Quality and Market Success of Android Applications," in *Proceedings - 2nd ACM International Conference on Mobile Software Engineering and Systems, MOBILESoft 2015*, 2015, pp. 22–32.

[17] P. D. Anjali, "Empirical Validation of Website Quality Using Statistical and Machine Learning Methods," pp. 1–6, 2014.

[18] D. D. J. Suwawi, E. Darwiyanto, and M. Rochmani, "Evaluation of academic website using ISO/IEC 9126," *2015 3rd Int. Conf. Inf. Commun. Technol. ICoICT 2015*, pp. 222–227, 2015.

[19] R. Mirsalari and P. N. Robillard, "Industrial Validation of an Approach to Measure Software Quality," *Sede*, p. 6, 2015.

[20] L. Kumar and S. K. Rath, "Hybrid functional link artificial neural network approach for predicting maintainability of object-oriented software," *J. Syst. Softw.*, vol. 121, pp. 170–190, 2016.

[21] S. Chakrabarty and N. Chaki, "Quality Evaluation of Conceptual Level Object Multidimensional Data Model," *Int. J. Comput. Appl.*, vol. 32, no. 3, p. 14, 2011.

[22] M. Pušnik, M. Heričko, Z. Budimac, and B. Šumak, "XML schema metrics for quality evaluation," *Comput. Sci. Inf. Syst.*, vol. 11, no. 4, pp. 1271–1290, 2014.

[23] Carnegie Mellon University., "Software Engineering Institute (SEI)," 1984. [Online]. Available: www.sei.cmu.edu.

[24] ISO/IEC, "ISO/IEC 25000 - Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE)," 2005.

[25] E. H. Marinho and R. F. Resende, "Quality Factors in Development Best Practices," in *ICCSA*, 2012, pp. 632–645.

[26] ISO/IEC, "INTERNATIONAL STANDARD ISO/IEC 25010:2010," 2010.

[27] ISO/IEC, "INTERNATIONAL STANDARD ISO/IEC DIS 25021 — Quality Measure Element," 2011.

[28] T. Punter, M. Ciolkowski, B. Freimut, I. John, F. Iese, and D.- Kaiserslautern, "Conducting On-line Surveys in Software Engineering Characterizing surveys in SE," 2003.

[29] T. C. Lethbridge, S. E. Sim, and J. Singer, "Studying Software Engineers : Data Collection Techniques for Software Field Studies," *Empir. Softw. Eng.*, vol. 10, pp. 311–341, 2005.

[30] B. Kitchenham and S. L. Pfleeger, "Principles of Survey Research Part 5: Populations and Samples," *ACM SIGSOFT Softw. Eng. Notes*, vol. 27, no. 5, p. 17, 2002.

[31] B. a Kitchenham and S. L. Pfleeger, "Principles of Survey Research Part 3: Constructing a Survey Instrument," *ACM SIGSOFT Softw. Eng. Notes*, vol. 27, no. 2, p. 20, 2002.

[32] B. Kitchenham and S. L. Pfleeger, "Principles of survey research part 4: questionnaire evaluation," *ACM SIGSOFT Softw. Eng. Notes*, vol. 27, no. 3, p. 20, 2002.

[33] B. Kitchenham and S. L. Pfleeger, "Principles of survey research part 6: Data Analysis," *ACM SIGSOFT Softw. Eng. Notes*, vol. 28, no. 2, p. 24, 2003.

[34] B. Kitchenham and S. L. Pfleeger, "Principles of Survey Research," *ACM SIGSOFT Softw. Eng. Notes*, vol. 26, no. 6, p. 16, 2001.

[35] B. a Kitchenham and S. L. Pfleeger, "Principles of Survey Research Part 2 : Designing a Survey Sample size Experimental designs," *Softw. Eng. Notes*, vol. 27, no. 1, pp. 18–20, 2002.

[36] H. Al-Kilidar, K. Cox, and B. Kitchenham, "The use and usefulness of the ISO/IEC 9126 quality standard," in *International Symposium on Empirical Software Engineering.*, 2005, pp. 122–128.

[37] R. E. Al-qutaish, "A Maturity Model of Software Product Quality," *Res. Pract. Inf. Technol.*, vol. 43, no. 4, pp. 307–328, 2010.

[38] E. Van Veenendaal, R. Hendriks, and R. Van Vonderen, “Measuring Software Product Quality,” *SQP*, vol. 5, no. 1, pp. 6–13, 2002.

[39] C. Wohlin, P. Runeson, M. Host, Magnus C. Ohlsson, Bjorn Regnell, and Anders Wesslen, *Experimentation in Software Engineering*. 2012.

[40] P. Torino and P. Torino, “A State-of-the-Practice Survey of Risk Management in Development with Off-the-Shelf A State-of-the-Practice Survey on Risk Management in Development with Off- The-Shelf Software Components,” vol. 34, no. April, pp. 271–286, 2016.

[41] A. Nugroho and M. R. V. Chaudron, “A survey into the rigor of UML use and its perceived impact on quality and productivity,” *Proc. Second ACM-IEEE Int. Symp. Empir. Softw. Eng. Meas. - ESEM ’08*, p. 90, 2008.

[42] J. Soini, “A Survey of Metrics Use in Finnish Software Companies,” *2011 Int. Symp. Empir. Softw. Eng. Meas.*, pp. 49–57, 2011.

[43] W. Chen, J. Li, J. Ma, R. Conradi, J. Ji, and C. Liu, “A survey of software development with open source components in Chinese software industry,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 4470 LNCS, pp. 208–220, 2007.

[44] J. Li, J. Ma, R. Conradi, W. Chen, J. Ji, and C. Liu, “A survey on the business relationship between Chinese outsourcing software suppliers and their outsourcers,” *Proc. - Asia-Pacific Softw. Eng. Conf. APSEC*, pp. 470–477, 2007.

[45] E. Petrinja, A. Sillitti, and G. Succi, “Adoption of OSS Development Practices by the Software Industry: A Survey,” *Open Source Syst. Grounding Res.*, vol. 365, pp. 233–243, 2011.

[46] A. Macphail, T. Hainey, and T. M. Connolly, “Applying Mlearning in Software Engineering Education : a Survey of Mobile Usage,” *Mob. Learn.*, no. iii, 2012.

[47] M. Usman, E. Mendes, and J. Börstler, “Effort estimation in Agile software development: A survey on the state of the practice,” *ACM Int. Conf. Proceeding Ser.*, vol. 27–29–Apri, 2015.

[48] R. N. Memon, S. S. Salim, and R. Ahmad, “Identifying research gaps in requirements engineering education: An analysis of a conceptual model and survey results,” *2012 IEEE Conf. Open Syst.*, pp. 1–6, 2012.

[49] Y. Cerqueira, S. R. De Lemos, Y. C. . Cavalcanti, P. A. . Da Mota Silveira Neto, I. . Do Carmo Machado, E. S. . De Almeida, and S. R. . De Lemos Meira, “Towards Understanding Software Change Request Assignment : a survey with practitioners,” *ACM Int. Conf. Proceeding Ser.*, pp. 195–206, 2013.

[50] C. Palomares, C. Quer, and X. Franch, “Requirements reuse and requirement patterns : a state of the practice survey,” *Empir. Softw. Eng.*, 2016.

[51] David Ameller, M. Galster, P. Avgeriou, and X. Franch, “A survey on quality attributes in service-based systems,” *Softw. Qual. J.*, vol. 24, no. 2, pp. 337–364, 2016.

[52] H. Tsuji, A. Sakurai, K. Yoshida, A. Tiwana, and A. Bush, “Questionnaire-Based Risk Assessment Scheme for,” pp. 114–127.

[53] H.-C. Huang, “Freemium business model: construct development and measurement validation,” *Internet Res.*, vol. 26, no. 3, pp. 604–625, 2016.

[54] D. Tofan, M. Galster, P. Avgeriou, and D. Weyns, “Software engineering researchers’ attitudes on case studies and experiments: An exploratory survey,” *Eval. Assess. Softw. Eng. (EASE 2011), 15th Annu. Conf.*, no. 638, pp. 91–95, 2011.

[55] O. Albayrak, “Instructor’s Acceptance of Games Utilization in Undergraduate Software Engineering Education: A Pilot Study in Turkey,” *2015 IEEE/ACM 4th Int. Work. Games Softw. Eng.*, pp. 43–49, 2015.

[56] D. Budgen, B. A. Kitchenham, S. M. Charters, M. Turner, P. Brereton, and S. G. Linkman, “Presenting software engineering results using structured abstracts: A randomised experiment,” *Empir. Softw. Eng.*, vol. 13, no. 4, pp. 435–468, 2008.

[57] F. Q. B. Da Silva and A. C. C. Frana, “Towards understanding the underlying structure of motivational factors for software engineers to guide the definition of motivational programs,” *J. Syst. Softw.*, vol. 85, no. 2, pp. 216–226, 2012.

[58] I. Erfurth and W. R. Rossak, “A look at typical difficulties in practical software development from the developer perspective A field study and a first solution proposal with UPEX,” *Proc. Int. Symp. Work. Eng. Comput. Based Syst.*, pp. 241–248, 2007.

[59] N. Fenton, M. Neil, W. Marsh, P. Hearty, Ł. Radliński, and P. Krause, “On the effectiveness of early life cycle defect prediction with Bayesian nets,” *Empir. Softw. Eng.*, vol. 13, no. 5, pp. 499–537, 2008.

[60] I. Garcia, C. Pacheco, and P. Sumano, “Use of questionnaire-based appraisal to improve the software acquisition process in small and medium enterprises,” *Stud. Comput. Intell.*, vol. 150, pp. 15–27, 2008.

[61] J. Hutchinson, J. Whittle, M. Rouncefield, and S. Kristoffersen, “Empirical assessment of MDE in industry,” *2011 33rd Int. Conf. Softw. Eng.*, pp. 471–480, 2011.

[62] M. V. Kosti, R. Feldt, and L. Angelis, “Personality, emotional intelligence and work preferences in software engineering: An empirical study,” *Inf. Softw. Technol.*, vol. 56, no. 8, pp. 973–990, 2014.

[63] T. Sedano, “Code readability testing, an empirical study,” *Proc. - 2016 IEEE 29th Conf. Softw. Eng. Educ. Training, CSEEandT 2016*, pp. 111–117, 2016.

[64] P. . Diebold, A. . Vetró, and D. . Méndez Fernández, “An Exploratory Study on Technology Transfer in Software Engineering,” *Int. Symp. Empir. Softw. Eng. Meas.*, vol. 2015–Novem, pp. 86–95, 2015.

[65] A. Forward and T. C. Lethbridge, “Problems and Opportunities for Model-Centric Versus Code-Centric Software Development: A Survey of Software Professionals,” *Proc. 2008 Int. Work. onModels Softw. Eng.*, pp. 27–32, 2008.

[66] J. Ji, J. Li, and R. Conradi, “Some lessons learned in conducting software engineering surveys in China,” *ESEM’08 Proc. 2008 ACM-IEEE Int. Symp. Empir. Softw. Eng. Meas.*, pp. 168–177, 2008.

[67] J. M. Rojas, G. Fraser, and A. Arcuri, “Automated Unit Test Generation During Software Development: A Controlled Experiment and Think-aloud Observations,” pp. 338–349, 2015.

[68] P. Karpati, Y. Redda, A. L. Opdahl, and G. Sindre, “Comparing attack trees and misuse cases in an industrial setting,” *Inf. Softw. Technol.*, vol. 56, no. 3, pp. 294–308, 2014.

[69] J. P. Campos, J. L. Braga, A. M. de Resende, and C. H. Os’orio Silva, “Identification of Aspect Candidates by Inspecting Use Cases Descriptions,” *SIGSOFT Softw. Eng. Notes*, vol. 35, no. 4, pp. 1–9, 2010.

[70] L. Prechelt and M. Liesenberg, “Design patterns in software maintenance: An experiment replication at Freie Universit??t Berlin,” *Proc. - 2011 2nd Int. Work. Replication Empir. Softw. Eng. Res. RESER 2011*, pp. 1–6, 2012.

[71] M. Haddara and A. Elragal, “ERP adoption cost factors identification and classification: a study in SMEs,” *Int. J. Inf. Syst. Proj. Manag.*, vol. 1, no. 2, pp. 5–21, 2013.

[72] A. Jedlitschka, “Evaluating a model of software managers’ information needs,” *Proc. 2010 ACM-IEEE Int. Symp. Empir. Softw. Eng. Meas. - ESEM ’10*, p. 1, 2010.

[73] M. Sensalire, P. Ogao, and A. Telea, “Evaluation of software visualization tools: Lessons learned,” *2009 5th IEEE Int. Work. Vis. Softw. Underst. Anal.*, pp. 19–26, 2009.

[74] M. Schmidberger and B. Brugge, “Need of Software Engineering Methods for High Performance Computing Applications,” *Parallel Distrib. Comput. (ISPDC), 2012 11th Int. Symp.*, pp. 40–46, 2012.

[75] D. Galin, *Software Quality Assurance From theory to implementation*. 2004.

[76] S. H. Kan, *Metrics and Models in Software Quality Engineering*. Addison-Wesley Longman Publishing Co., Inc., 2002.

[77] Rohaiza Abd. Rokis, “Youth Employability and Work Attitudes,” *Int. J. Sci. Commer. Humanit.*, vol. 2, no. 5, p. 13, 2014.

[78] G. Vigderhous, *The Level of Measurement and Permissible Statistical Analysis in Social Research*, vol. 20, no. 1. 1977.

[79] U. Jakobsson, “Statistical presentation and analysis of ordinal data in nursing research,” *Scand. J. Caring Sci.*, vol. 18, no. 4, pp. 437–440, 2004.

[80] IEEE Computer Society, “IEEE Standard for a Software Quality Metrics Methodology - IEEE Std 1061™-1998 (R2009),” vol. 1998, 2009.

[81] H. Jung and S. Kim, “Measuring Software Product Quality: A Survey of ISO/IEC 9126,” *IEEE Softw.*, pp. 88–92, 2004.

[82] L. M. Lozano, E. García-Cueto, and J. Muñiz, “Effect of the Number of Response Categories on the Reliability and Validity of Rating Scales,” *Methodol. Eur. J. Res. Methods Behav. Soc. Sci.*, vol. 4, no. 2, pp. 73–79, Jan. 2008.

[83] C. Alzola and F. Harrell, “An Introduction to S and the Hmisc and Design Libraries,” 2006. [Online]. Available: http://her.gr.distfiles.macports.org/mirrors/CRAN/doc/contrib/Alzola+Harrell-Hmisc-Design-Intro.pdf. [Accessed: 29-Jan-2015].

[84] “Technology Evaluation Centers.” [Online]. Available: http://www.technologyevaluation.com/. [Accessed: 18-Apr-2017].

[85] J. Baroudi, M. Olson, and B. Ives, “An empirical study of the impact of user involvement on system usage and information satisfaction,” *Commun. ACM*, vol. 29, no. 3, pp. 232–238, 1986.

[86] Software Engineering Institute, “CMMI® for Development, Version 1.3,” 2010.

[87] IEEE Computer Society, *SWEBOK Guide*. 2014.

[88] S. Hansen and J. Rennecker, “Getting on the same page: Collective hermeneutics in a systems development team,” *Inf. Organ.*, vol. 20, no. 1, pp. 44–63, Jan. 2010.

[89] O. Braud, “Facteurs decisionnels pour l’iplantation d’un ERP dans les PME : Le role de l’evaluation des benefices tangibles et intangibles,” 2008.

[90] A. J. Albrecht, “Measuring application development productivity,” *IBO Conf. Appl. Dev.*, vol. 10, pp. 83–92, 1979.

[91] R. Alvaro, “Framework for a global quality evaluation of a website,” *Online Inf. Rev.*, vol. 36, no. 3, pp. 347–382, 2012.

[92] H. Yang, “Measuring software product quality with ISO standards base on fuzzy logic technique,” *Adv. Intell. Soft Comput.*, vol. 137 AISC, pp. 59–67, 2012.

[93] P. Tomas, M. J. Escalona, and M. Mejias, “Open source tools for measuring the Internal Quality of Java software products. A survey,” *Comput. Stand. Interfaces*, vol. 36, no. 1, pp. 244–255, 2013.

[94] M. Kwiatkowski and C. Verhoef, “Recovering management information from source code,” *Sci.*

Comput. Program., vol. 78, no. 9, pp. 1368–1406, 2013.

[95] S. Lehtonen, “Metrics for Gerrit code reviews,” no. May, pp. 31–45, 2015.

[96] B. H. Layne, J. R. Decristoforo, and D. Mcginty, “Electronic versus traditional student ratings of instruction,” *Res. High. Educ.*, vol. 40, no. 2, pp. 221–232, 1999.

[97] B. Kitchenham, O. Pearl Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, “Systematic literature reviews in software engineering - A systematic literature review,” *Inf. Softw. Technol.*, vol. 51, no. 1, pp. 7–15, 2009.

[98] Ameller David, Galster Matthias, Paris Avgeriou, and Franch Xavier, “A survey on quality attributes in service-based systems,” *Softw. Qual J*, 2016.

[99] K. Dullemond and B. van Gameren, “What Distributed Software Teams Need to Know and When: An Empirical Study,” *Glob. Softw. Eng. (ICGSE), 2013 IEEE 8th Int. Conf.*, pp. 61–70, 2013.

[100] J. M. Rojas, G. Fraser, and A. Arcuri, “Automated unit test generation during software development: a controlled experiment and think-aloud observations,” *Proc. 2015 Int. Symp. Softw. Test. Anal. - ISSTA 2015*, pp. 338–349, 2015.

[101] A. Jedlitschka, “Evaluating a model of software managers’ information needs,” *Proc. 2010 ACM-IEEE Int. Symp. Empir. Softw. Eng. Meas. - ESEM ’10*, p. 1, 2010.

[102] A. Nugroho and C. F. J. Lange, “On the relation between class-count and modeling effort,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 5002 LNCS, pp. 93–104, 2008.

APPENDIX B – ARTICLE 2: EXPECTED SOFTWARE QUALITY PROFILE: A METHODOLOGY AND A CASE STUDY

Reza Mirsalari
Laboratoire de génie logiciel
École Polytechnique de Montréal
Montréal, Canada
reza.mirsalari@polymtl.ca

Pierre N. Robillard
Laboratoire de génie logiciel
École Polytechnique de Montréal
Montréal, Canada
pierre-n.robillard@polymtl.ca

Accepted at IEMCON 2016 conference

For decades, the notion of software quality evaluation is raised as a challenging task. Recently many studies have presented quality evaluation methodologies for specific domains or specific techniques. They usually select a pre-defined model, customize the characteristics, define the metrics and evaluate the quality of the product or development process. Our study presents a bottom-up methodology for the quality evaluation process. In this paper, we present a methodology to create the expected quality profile. In our approach, the first step is listening to the users, and then retrieving the most important quality factors and creating a model to evaluate the expected quality of the software product. The profile is formed by eliciting the expected users' quality expectations, and then quantifying the elicited factors by applying them to our quality evaluation model and the ISO/IEC 25000 standard. The result of this research empowers the software development stakeholders to perform a crosscheck between users' specific quality expectations and other drivers (functional and architecture/design requirements), before or during the software development process. The crosscheck aims to guarantee that there are enough activities, roles and artifacts in the software development process to support the users' quality requirements.

Keywords—software engineering; quality assurance; expected quality; standard

Introduction

Interest in software quality goes back to the early age of software development. Nowadays, software quality is fundamental to software success [85]. That is, failure in software products may cause major financial loss [36][37]. While attention to the software's functional specifications is important and necessary, the software's quality characteristics should also be taken into consideration given their impact on products, projects, processes and people.

A software product may appear to work well; the programming may be complete; it can be successfully installed at the client's site, and it can serve thousands of users worldwide. But, generally:

- it fails for short periods of time;
- there are poor failure-detection features;
- the absence of a comprehensive technical manual requires programmers to spend unplanned time dealing with bugs or treat minor software changes;
- new users may be required to spend undue time learning how to work with unintuitive features.

These examples illustrate that although software products may adequately meet functional and installation requirements, they may still suffer from poor performance in important quality areas such as maintenance, reliability, software reuse, or training/uptake.

Poor performance on quality measures is often the result of anomalies that creep in during the development process, whether on the part of developers, managers, or customers – primarily in the form of code, procedure, documentation, or data errors. [75]. Consequently, software quality should be monitored and controlled during the development process and while being deployed to the end users.

However, software quality faces some less obvious challenges and influences. First, software quality depends on the usage context. The quality factors of a network application are different from those of a medical firmware or game software. Second, the stakeholders related to a software product can vary. For example, the users of a tax management software vary widely: they may be manufacturers, school teachers, or physicians, in addition to the power users and software maintenance team members. This variety of stakeholders is potentially problematic for quality evaluation because different stakeholders have different perspectives and different quality requirements, sometimes paradoxical. Furthermore, the

literature and standards are not 100% clear and consistent. The terms and concepts are interpretable. For example, the terms "quality", and "performance" have different meanings in different documents [86][87]. Lastly, the diversity of data owners increases the risk of diversity in interpretations and the high cost of data collection may make quality evaluation challenging [88].

In the ever-evolving software market, software buyers face a central challenge: Among these various products, which one best meets my requirements *and* my budget? While most software buyers are aware of their functional and budgetary requirements, quality factors such as availability or reliability are not usually taken into account. Software vendors rarely talk about the quality aspects of their products [89].

In this research, we aimed to create the expected quality profile that is also called Quality Plan Profile. Creating the Quality Plan Profile includes eliciting the expected users' quality requirements, and then quantifying the elicited quality factors by applying them to our quality evaluation model and the ISO/IEC 25000 series of standard.

The rest of this paper is organized as follows. In section II we present a literature review in addition to a brief explanation of ISO/IEC 25000 as the basis of our research. In section III we discuss our methodology for creation and refinement the quality questionnaire and applying the questionnaire to the standard quality characteristics. In section IV, the obtained results and the analysis of the data received from the accomplishment of the survey are discussed. In section V, we conclude the results of the research.

Literature review

Software quality evaluation is not a new notion in software engineering literature. The studies on software quality evaluation goes back to the 70s and afterwards when the leading integrated hierarchy models such as McCall (1977), Boehm (1978), Grady (1992), Dromey (1996), and recently ISO/IEC 9126 (2001) and ISO/IEC 25000 (2001) presented a list of quality factors, relationships, and methodologies for quality evaluation. There are other types of quality models that evaluate specific quality characteristics, such as Albrecht function point model [90], exponential distribution, reliability growth model and Rayleigh model [76], which are generally designed to evaluate the reliability of the software.

Moreover, recently many studies have presented quality evaluation methodologies for specific domains or techniques. Rocha [91] proposes a high-level model for a global quality evaluation of a website. Yang [92] proposes a methodology to evaluate the software quality using the fuzzy logic technique. There are a lot of studies that analyze the software source code for evaluating the internal quality factors, such as the reports in [93], [94], [95].

The quality evaluation models in the current literature, usually attempt to evaluate the software quality using a top-down approach. They select a pre-defined model, customize the characteristics, define the metrics and evaluate the quality of the product or development process. Our study presents a bottom-up methodology. In our approach, the first step is listening to the users, and then retrieving the most important quality factors and creating a model to evaluate the expected quality of the software product.

This research is fundamentally based on quality models and the quality characteristics presented in ISO/IEC 25000 [24]. For creating our quality evaluation model, a questionnaire was created in order to elicit the user's quality expectations and to create the quality plan profile. ISO/IEC 25000 aims to create a framework for evaluating the software product quality. The product quality concept in ISO/IEC 25000 has been presented in five divisions: Quality management, Quality model, Quality measurement, Quality requirement, and Quality evaluation. In our research, we emphasize on Quality model, Quality measurement, and quality evaluation divisions; i.e. ISO/IEC 25010, ISO/IEC 25021, and ISO/IEC 25040.

The ISO/IEC 25010 looks at the product quality from two viewpoints: "quality in-use" model composed of five characteristics (Effectiveness, Efficiency, Satisfaction, Freedom from Risk, Context Coverage) that relate to the outcome of interaction when a product is used in a particular context of use, and "product quality" model composed of eight characteristics (Functional Suitability, Performance Efficiency, Compatibility, Usability, Reliability, Security, Maintainability, Portability) that relate to static and dynamic properties of the software product. Some of the characteristics are further decomposed into sub-characteristics.

ISO/IEC 25021 provides guides to specify Quality Measure Elements (QME). QME is a measure defined in terms of a property and the measurement method for quantifying it, including optionally the transformation by a mathematical function. Quality measures are formed from applying the QMEs in a Measurement Function (Figure 20). The quality measures are then associated with the standard quality characteristics and sub-characteristics. [24].

Figure 26: Standard Product Quality Model – ISO/IEC 25000

Methodology

In this study, the stakeholders' quality expectations are investigated and identified using a structured working questionnaire. The questionnaire consists of a number of questions regarding product quality characteristics. These characteristics are intended to be expressed in the stakeholders' language. The provided answers are utilized to recognize the most relevant quality characteristics and the associated values. In the following, the steps of creating, revising the questionnaire and finalizing the quality evaluation model are explained.

Questionnaire Creation

At the initial step, a set of preliminary information in question form was elicited. In this step, using a free and unfiltered search, the researchers identified 152 questions related to quality measure elements. These preliminary questions were found by reviewing the related literature such as [75], [76], [2], websites, working instructions, norms, standards, and any other resources. It is worth mentioning that although this process is totally a non-systematic review, our methodology has the capacity to be initiated by any set of quality related questions. Since the questions will be refined in the next stages, they can be extracted non-systematically using a wild literature review, or using a systematic review for a specific context.

In addition to the questions, eleven different rating scales (Table 42) are also developed and assigned to each question.

TABLE 42: RATING SCALES

Title	***Scale***
Absolute Yes/No	Absolutely No, Yes - for 20% to 50% of cases, Yes - for 50% to 80% of cases, Absolutely Yes
100% - 0% of time	75% - 100% of time, 50% - 75% of time, 25% - 50% of time, 0% - 25% of time
1h - 1w	Up to 1 hour, 1 hour to 24 hours, 1 day to 1 week, More than 1 week
High-Low Probability	High (70%-100%), Medium (30%-70%), Low (0%-30%)
All-None	None, Least, Most, All
Agree-Disagree	Disagree, Mostly agree, Completely agree
Importance	Absolutely Essential, Very Important, Of Average Importance, Of Little Importance, Not Important at All
Yes/No	Yes, No
Numeric	Number 1..10,000
Poor - Excellent	Poor, Fair, Good, Very Good, Excellent
Support	Unsupported, Poorly supported, Fairly supported, supported

Questionnaire Refinement

In this step, the questions are refined using an iterative approach. The goal of the question refinement step is to make the questionnaire more reliable and valid. This step was done by adapting a Delphi method in three iterations. Delphi [2] is a systematic iterative communication technique that aims to increase the consensus among the members of a panel of experts. This step was done by a panel of 7 experts in a software acquisition consultation company. The researcher acted as a coordinator. The expert team was asked to review the questions and determine if the questions and their rating scales are appropriate and relevant. The experts, who were selected by the company's chief technical officer, are staff with more than ten years of experience in software acquisition, enterprise software implementation, and deployment. The experts were also required to validate the questions by providing one of the pre-defined feedback types on the relevance of each of the questions.

As stated, the questionnaire was improved through an iterative approach. The purpose of the first iteration is to figure out how reliable the questions are, and how the ambiguities among the questions can be discovered and resolved. The reliability checking was done by asking the experts to review the questions. Accordingly, the questionnaire was conveyed to the panel of experts. The experts reviewed the questions and sent their feedback and comments to the coordinator. The coordinator applied the feedbacks to the questions and generated the composite report. The activities in the first iteration resulted in a reduction of 152 questions to 125.

The purpose of the second iteration is to find out how valid the modified questions are, and also to find out if the modifications in the first iteration have decreased the ambiguities and increased the convergence among the experts. For this purpose, six experts were asked to review the composite report and send their feedback and comments as Agree or Disagree to the coordinator. Thus, the main question that was asked of the experts in this iteration is "Do you agree with the modifications made for each question?" The items that were expected to be done by the experts in the second iteration are: 1) read the decision made for each question, 2) read the updated question and rating method, 3) read the comments of the other experts for each question, and 4) state your final opinion whether you agree or disagree with the final decision.

After receiving the feedbacks from the experts, we counted the Agreed and Disagreed answers and compared it with the obtained agreement value from the first iteration. The result shows that on average, the agreement value increased by 20%; i.e. from 58% to 78%. Applying the feedback also resulted in decreasing the number of questions from 125 to 106.

In the third iteration, the outlier questions were removed and also some questions were rephrased and polished to have only one type of rating scale for all the questions. As it was mentioned earlier, eleven rating scales were defined to categorize the questionnaire responses. These scales include both *ordinal* scales such as "Strongly disagree", "Disagree", "Agree", "Strongly agree", and *absolute* scales such as the integer values that are assigned to "number of help desk calls during one year of service". For more clarification, three sample questions with their rating scales are presented in Table XXXIV.

TABLE 43: SAMPLE QUESTIONS AND RATING SCALES

	Question	*Rating Scale*
Before scales unification	1. Is there a Help that helps, and matches the functionality?	Yes/No
	2. Are there features to distinguish mandatory fields?	Absolute Yes/No
	3. What proportion of potential users chooses to actually use the system?	All-None
After scales unification	1. The capability of the Help function within the product to provide adequate guidance on most issues.	Importance
	2. The capability of the product to present a feature to distinguish mandatory fields.	Importance
	3. The enjoyment of the users while using the product and feel fully engaged with it.	Importance

Comparing the data with different scales is problematic. To simplify this comparison, it is needed to unify all the various scales. For this purpose, we rephrased the questions to obtain one single scale that is "Importance".

The other activity in the third iteration was removing the outliers from the questionnaire. We define the outliers as the controversial questions; i.e. the questions that were not approved by the majority of the experts. Consequently, we kept the questions that were approved by at least five experts (out of six) and removed the rest. In this way, 25 questions were removed, and 91 questions remained.

At the end of the third iteration, all the questions were approved by the industrial experts. Although the experts are all professionals in software acquisition, we asked two academic professors, who are professional in software quality and the user interface domain, to review the questionnaire. The goal of this complementary iteration is to review and validate the questionnaire from the theoretical, scientific, and academic point of view. The academic reviewers recommended removal of some marginal questions and only keep the ones that are focused on the software quality subjects.

By the result of this complementary iteration, we came up with 50 final questions. This set of questions are those that are validated and approved by both industrial and academic professionals. In Table XXXV a sample of the final questions for the end and the power users, and the shared questions for both groups are presented.

According to the comments received from the academic professionals, we separated the questionnaire for power users and end users. We define end users as the persons who regularly work with the software application for data entry and report generation. The power users are defined as the technical, knowledgeable professionals who have been granted high-level access rights for the software administration and control. Thus, the end user's questionnaire includes the quality aspects closer to the quality in-use, and the power users' questionnaire consists of the questions more related to product quality defined in ISO/IEC 25000. The end user questionnaire includes 37 questions and power user questionnaire includes 33 questions. The end user's questionnaire is constructed from 17 questions specifically dedicated to the end users, plus 20 questions shared between the both user groups. The Power user's questionnaire is constructed from 13 questions specifically dedicated to Power users, plus 20 questions shared between the both user groups. In total 50 distinct questions are distributed in two questionnaires (17+20+13=50).

TABLE 44: SAMPLE FINAL QUESTIONS

User Group	*Question*
End User	Error handling agility: The capability of the product to handle data-input errors quickly and easily.
	Learnability: The capability of the product to be learned easily and quickly.
Power User	Tools for maintenance: The availability of the development tools for the technical power users to add features in the future.
	Stress Handling: The capability of the product to cope when exceeding various limits.
Both	Backward compatibility: The capability of the product to use the files and data created with older versions.
	Responsiveness: The capability of the software to perform most functions at the acceptable speed.

Association with the Standard: A Coding Activity

The questions were associated with the quality measures by the researcher, and the quality measures were associated with the quality characteristics that are presented in ISO 25010. In this step, the concepts and the keywords of each question were retrieved, and then the questions were associated to at least one of the standard product quality characteristics. Since one question may address more than one quality aspects, the relationship between the questions and the quality characteristics is a "many to many" relationship. This step resulted in creating a matrix of associations (Model matrix) with the questions in the rows and the quality characteristics in the columns. Each element of the Model matrix is marked ('X') if the given question in the row is associated with the quality characteristic in the column. In Table XXXVI, a small part of a hypothetical Model matrix is presented. The marks in this matrix show that the question Q1 is associated with *Effectiveness* and *Efficiency*, the Q2 is associated with *Efficiency* and *Satisfaction* and Q3 is associated with *Effectiveness* and *Satisfaction*.

Table 45: Elements of a hypothetical Model matrix

	Effectiveness	*Efficiency*	*Satisfaction*
Q1	X	X	
Q2		X	X
Q3	X		X

As a sample, the diagram in Figure 21 shows the number of questions associated with the quality characteristics, for end users in quality in-use.

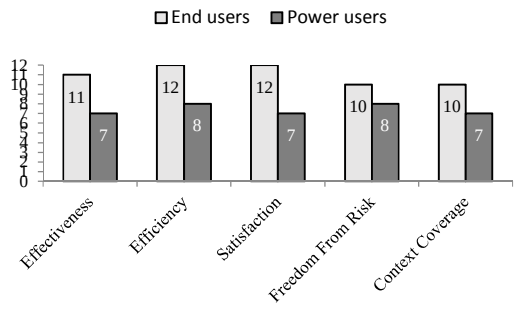


Figure 27: Number of questions associated with quality in-use characteristics for end users

All the questions in both end users' and power users' questionnaires are associated to 2, 3, or 4 quality characteristics as it is illustrated in Figure 22. The Y-axis in Figure 22 shows the question ids and the X-axis shows the number of quality characteristics associated with each question.

Case Study: A Survey

In order to practically validate the quality evaluation model, we targeted the probable users of a software product. Using waterfall methodology, a development team attempts to create an administration and accounting software product for collegiate affairs. We applied our quality evaluation model to the software product by selecting a set of end users and power users, and asking them to participate in a survey.

By asking 17 participants, including 9 end users and 8 power users, the survey was accomplished. The end users and the power users answered their respective questionnaire. The two questionnaires for end and power user were implemented in the Google Forms application. Besides, an "Ethics Compliance Certificate" was also issued for this project. The anonymous responses were received and the questionnaires were closed after 15 days. In the following, the analysis process of the results are explained.

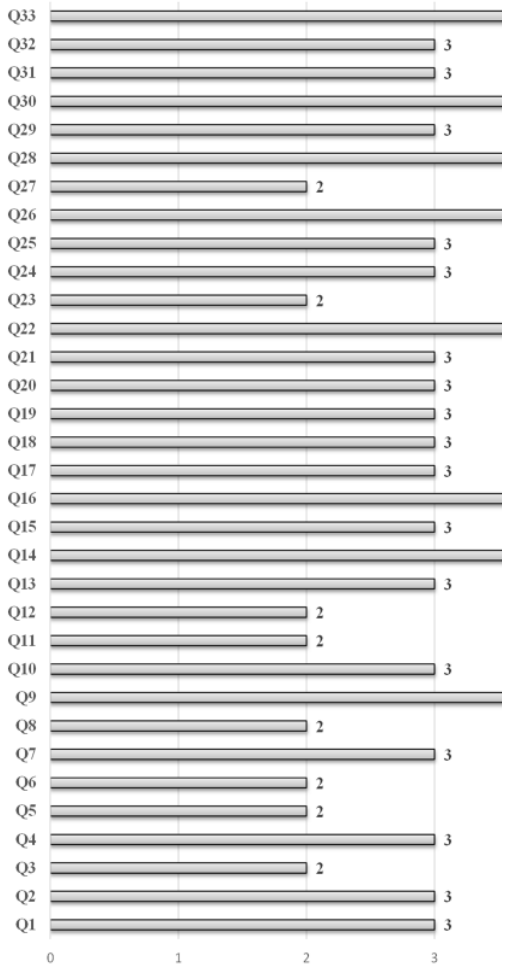


Figure 28: Number of quality characterisitics related to each question – end user's questionnaire

Analysis of the Responses

The analysis of nominal, interval and ratio scales seems to be straightforward and transparent while it is not for ordinal scales [77]. In a variety of applied fields, the level of measurement and the adequacy of treating ordinal data as interval data continues to be controversial [78] [79]. Thus, we applied a "counting" approach for analyzing the responses of the surveys. As the first step, the ordinal scales are identified from A to E, as the following:

A: Absolutely Essentials

B: Very Important

C: Of Average Importance

D: Of Little Importance

E: Not Important at All

Then the number of obtained scales for each question is counted, and a count_string is formed to show the number of each scale for a given question. The count_strings facilitate the counting and analysis of the responses. For example, the count_string=A2B5C2 means we have received 2 Absolutely Essential, 5 Very Important, and 2 Of Average Importance scales for a given question. In the next step, we put the count_string values in place of the marked elements of the Model matrix. Consequently, we will have another matrix called Value matrix. For example, if the count_string for the question Q1 is calculated as B3C4D3, for Q2 as A5D2E5 and for Q3 as B4D5E7, we have a hypothetical Value matrix for the above Model matrix, as it is illustrated in Table 46.

Table 46: Element of a hypothetical Value matrix

	Effectiveness	***Efficiency***	***Satisfaction***
Q1	B3C4D3	B3C4D3	
Q2		A5D2E5	A5D2E5
Q3	B4D5E7		B4D5E7

At the end, the total of each scale of each quality characteristic is calculated. For instance, the data in Table 47 shows the number of each responded rating scale related to quality in-use for end users.

TABLE 47: NUMBERS AND PERCENTAGE OF EACH RATING SCALES: END USER QUESTIONNAIRE – QUALITY IN-USE

	Effectiveness	***Efficiency***	***Satisfaction***	***Freedom From Risk***	***Context Coverage***
↓ Five-Scale ↓					
A	31(33%)	47(36%)	26(20%)	24(31%)	12(18%)
B	32(34%)	45(34%)	43(33%)	32(42%)	19(28%)
C	25(27%)	29(22%)	44(34%)	12(16%)	24(36%)
D	2(2%)	4(3%)	6(5%)	4(5%)	2(3%)
E	4(4%)	6(55)	12(9%)	5(6%)	10(15%)
↓ Three-Scale ↓					
A+B	67%	70%	53%	72%	46%
C	27%	22%	34%	16%	36%
D+E	6%	8%	14%	12%	18%
Sum	***100%***	***100%***	***100%***	***100%***	***100%***

As it was mentioned earlier, 20 questions were shared in the two questionnaires for end and power users. We applied the Fisher's Exact Test to check if there is a significant difference between the end and power user's responses. The results of the Fisher's Test are shown in Table 48 . We defined the null hypothesis as "there is no significant difference between the two categories".

TABLE 48: THE *P-VALUE* RESULTING FROM FISHER'S TEST

Question ID	***p-value***	***Question ID***	***p-value***
Q1	1	Q11	0.61
Q2	1	Q12	1
Q3	1	Q13	0.52
Q4	0.55	Q14	0.26
Q5	0.78	Q15	0.26
Q6	1	Q16	0.46
Q7	1	Q17	0.64
Q8	0.87	Q18	0.46
Q9	1	Q19	0.40
Q10	0.43	Q20	0.82

Fisher's test calculates the *p-value*, to show if the null hypothesis can be rejected. If the *p-value* is less than 0.05 then the null hypothesis can be rejected, otherwise, the alternative hypothesis can be accepted. The value of *p-value* was calculated in R. The Fisher's tests are applied to the shared questions. Because the *p-values* are more than 0.05, we cannot reject the null hypothesis. In other words, we can say that the values are from the same distribution. This means that both end users and power users have the same belief about the importance of the certain quality factors, by using our questionnaires. In other words, the results of all *p-values* strongly show that there is no significant difference between the quality expectations of the both user groups.

Expected Quality Profile

We designed the expected quality profile as a three-scale percentage style diagram. We compressed the five-scale presentation to three-scale by:

Very Important ="Absolutely essential" + "Very important"
Neutral = "Of average importance"
Not Important = "Of Little Importance" + "Not Important at All"

The data of the expected quality profile for the end users in quality in-use characteristics are shown in Figure 30. The percentage stacks show that from the end users point of view, the *Effectiveness*, the *Efficiency*, and the *Freedom from Risk* are the most important quality characteristics, while they relatively find the *Context Coverage* a less important factor.

The similar profile can be created for product quality characteristics, as it is shown in Figure 31. The profile in Figure 31 includes the percentages of

importance for the product quality characteristics of end users. The same diagram can be created for the power users. The percentage stacks shown in Figure 31 implies that the *Security* and the *Reliability* are the most important characteristics from the end users' point of view while the *Portability* is the least important factor.

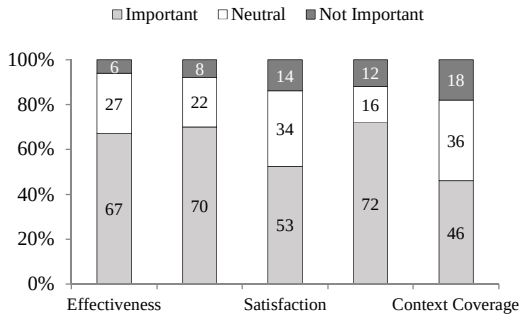


Figure 29: Expected quality profile - End users - Quality in-use

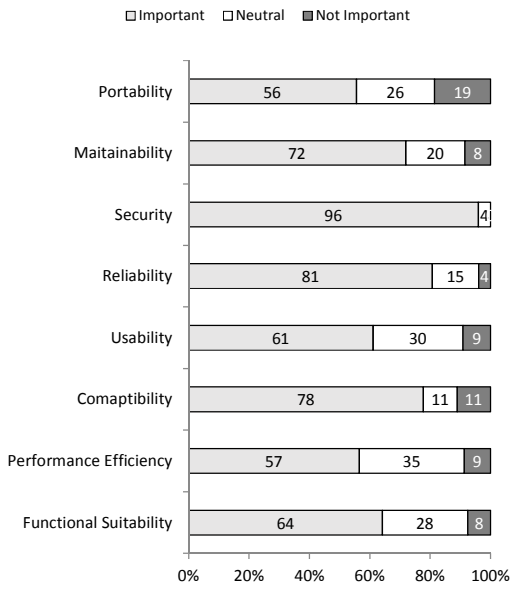


Figure 30: Expected Quality profile – End users - Product quality

Cause-Effect Analysis

The creation process of our quality evaluation model is a round trip movement that includes two stages: 1) forward construction that is a macro perspective, and 2) backward analysis that is a micro perspective. Until now, we constructed the questionnaires, gathered the data from the real users and created the high-level quality profiles. The next stage is to move backward to the details and see what causes the importance values shown on the profile diagrams. For answering this question, we need to do a filtration on the data addressing the questions that cause the profile. The tool that we used for this illustration is the Pareto analysis. The Pareto analysis is a technique that shows the percentage of the influencing factors that causes the final result.

For example, on the end user's quality profile, the importance value of the "Satisfaction" is 53% as shown in Figure 29. It would be helpful if we could know "what are the questions that caused the majority portion of the 53%". The Pareto analysis gives the answer. As it is illustrated in the Pareto chart in Figure 31, the question number Q30, Q19, Q20, Q16, Q29, Q8, Q33, Q32 have formed the basis for 84% of the importance. Thus, we can say if the software is able to answer the above questions, then it is able to "satisfy" 84% of the end user's quality expectations. The Pareto analysis can be applied to other quality characteristics on the profile for end users and power users. It should be noted that the "84%" is chosen only as an example. Any other value can be chosen as the boundary of the cause-effect analysis.

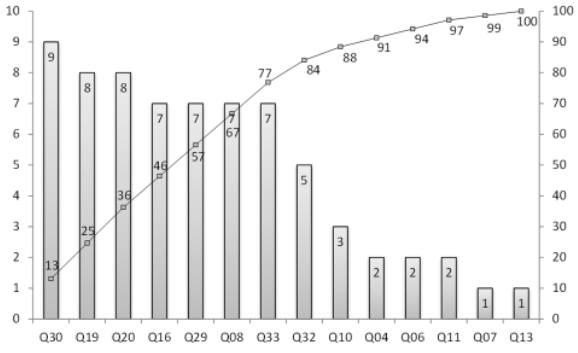


Figure 31: Pareto analysis for "Satisfaction" – End users

Conclusion

In this research, we aimed to present a methodology that enables the evaluation of the expected quality characteristics of a software product. For this purpose, we collected a set of questions related to the quality measures, which were reviewed iteratively by industry experts and academic professionals. The final questions formed two working questionnaires, one for end users and one for power users. A software product – in development phase – was selected as a case study and the future users of the software were asked to fill out the questionnaire while taking the software product and their requirements into consideration. Statistical analysis of the responses reveals that end users and power users have the same perspective about the importance of certain quality factors. The results show the feasibility of identifying and quantifying software users' quality-related expectations using the model presented in ISO/IEC 25000. Besides, the results of the case study show specifically that *efficiency* and

freedom from risk are the most important factors for both types of users in the quality in-use category. In the product quality category, the most important characteristics were *security* and *reliability*. Using the Pareto analysis, this research provides the tools for retrieving the causes that affect the importance of any quality characteristic from the users' perspective.

One of the factors that distinguish high-quality software product from a low-quality one is the degree of "user involvement", which is defined as the level of users' enthusiasm to engage with the software product. Users are generally most willing to engage with software products that not only meet their functional requirements, but also their quality expectations. The results of this research help frame the importance of quality as an essential field of focus during software development and evaluation, and the tools developed – the "quality plan profiles" – enable developers to tailor and specify quality-related requirements in a concrete way.

With quality plan profiles in hand, it is possible to perform a useful crosscheck between users' specific quality expectations and other drivers (functional and architecture/design requirements), before or during the software development process. The crosscheck should be aimed to guarantee that there are enough activities and sub-activities in the software development process to support the users' quality expectations.

Acknowledgements

Many thanks to all participants at Technology Evaluation Centers (TEC). Special thanks to the CTO of TEC, Mehdi Aftahi, and the participants in the Polytechnique Montréal for their continued support and helpful input. This work was funded in part by the Canadian Mitacs-Accelerate program, grant number IT06440.

References

[1] R. Mirsalari and P. N. Robillard, "Industrial Validation of an Approach to Measure Software Quality," in *SEDE*, 2015, p. 6.

[2] L. Westfall, *The Certified Software Quality Engineer Handbook*. ASQ Quality Press; Har/Cdr edition, 2009.

[3] I. Standard, "ISO/IEC Systems and Software Engineering—Vocabulary (ISO/IEC/IEEE 24765:2010," 2015.

[4] Crosby Philip B., *Quality is free*. McGraw-Hill Science/Engineering/Math; 3 edition, 1979.

[5] J. M. Juran, *Juran's Quality Handbook*, Fifth Edit. New York, USA: McGraw-Hill, 1999.

[6] G. Schulmeyer and J. McManus, *Handbook of Software Quality Assurance*, 3rd Ed. Upper Saddle River, NJ: Prentice Hall PTR, 1998.

[7] L. Arksey, H., & O'Malley, "Scoping studies: towards a methodological framework," *Int. J. Soc. Res. Methodol.*, vol. 8, no. 1, pp. 19–32, 2005.

[8] H. L. Colquhoun, D. Levac, K. K. O'Brien, S. Straus, A. C. Tricco, L. Perrier, M. Kastner, and D. Moher, "Scoping reviews: Time for clarity in definition, methods, and reporting," *J. Clin. Epidemiol.*, vol. 67, no. 12, pp. 1291–1294, 2014.

[9] R. Mirsalari and P. N. Robillard, "Expected Software Quality Profile: A methodology and a case study," in *The 7th IEEE Annual Information Technology, Electronics & Mobile Communication Conference - IEMCON 2016*, 2016, p. 923.

[10] M. Kläs, C. Lampasona, and J. Münch, "Adapting software quality models: Practical challenges, approach, and first empirical results," in *Proceedings - 37th EUROMICRO Conference on Software Engineering and Advanced Applications, SEAA 2011*, 2011, pp. 341–348.

[11] J. Gottschick and H. Reste, "An empirical evaluation of the quality of interoperability specifications for the web," in *Proceedings - 36th EUROMICRO Conference on Software Engineering and Advanced Applications, SEAA 2010*, 2010, pp. 398–405.

[12] I. Biscoglio and E. Marchetti, "An experiment of software quality evaluation in the audio-visual media preservation context," in *Proceedings - 2014 9th International Conference on the Quality of Information and Communications Technology, QUATIC 2014*, 2014, pp. 118–123.

[13] L. Aversano and M. Tortorella, "Analysing the Reliability of Open Source Software Projects," pp. 348–357, 2015.

[14] K. Lochmann, J. Ramadani, and S. Wagner, "Are comprehensive quality models necessary for evaluating software quality?," in *Proceedings of the 9th International Conference on Predictive Models in Software Engineering - PROMISE '13*, 2013, pp. 1–9.

[15] R. Hofman, "Behavioral economics in software quality engineering," *Empir. Softw. Eng.*, vol. 16, no. 2, pp. 278–293, 2011.

[16] L. Corral and I. Fronza, "Better Code for Better Apps: A Study on Source Code Quality and Market Success of Android Applications," in *Proceedings - 2nd ACM International Conference on Mobile Software Engineering and Systems, MOBILESoft 2015*, 2015, pp. 22–32.

[17] P. D. Anjali, "Empirical Validation of Website Quality Using Statistical and Machine Learning Methods," pp. 1–6, 2014.

[18] D. D. J. Suwawi, E. Darwiyanto, and M. Rochmani, "Evaluation of academic website using ISO/IEC 9126," *2015 3rd Int. Conf. Inf. Commun. Technol. ICoICT 2015*, pp. 222–227, 2015.

[19] R. Mirsalari and P. N. Robillard, “Industrial Validation of an Approach to Measure Software Quality,” *Sede*, p. 6, 2015.

[20] L. Kumar and S. K. Rath, “Hybrid functional link artificial neural network approach for predicting maintainability of object-oriented software,” *J. Syst. Softw.*, vol. 121, pp. 170–190, 2016.

[21] S. Chakrabarty and N. Chaki, “Quality Evaluation of Conceptual Level Object Multidimensional Data Model,” *Int. J. Comput. Appl.*, vol. 32, no. 3, p. 14, 2011.

[22] M. Pušnik, M. Heričko, Z. Budimac, and B. Šumak, “XML schema metrics for quality evaluation,” *Comput. Sci. Inf. Syst.*, vol. 11, no. 4, pp. 1271–1290, 2014.

[23] Carnegie Mellon University., “Software Engineering Institute (SEI),” 1984. [Online]. Available: www.sei.cmu.edu.

[24] ISO/IEC, “ISO/IEC 25000 - Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE),” 2005.

[25] E. H. Marinho and R. F. Resende, “Quality Factors in Development Best Practices,” in *ICCSA*, 2012, pp. 632–645.

[26] ISO/IEC, “INTERNATIONAL STANDARD ISO/IEC 25010:2010,” 2010.

[27] ISO/IEC, “INTERNATIONAL STANDARD ISO/IEC DIS 25021 — Quality Measure Element,” 2011.

[28] T. Punter, M. Ciolkowski, B. Freimut, I. John, F. Iese, and D.- Kaiserslautern, “Conducting On-line Surveys in Software Engineering Characterizing surveys in SE,” 2003.

[29] T. C. Lethbridge, S. E. Sim, and J. Singer, “Studying Software Engineers : Data Collection Techniques for Software Field Studies,” *Empir. Softw. Eng.*, vol. 10, pp. 311–341, 2005.

[30] B. Kitchenham and S. L. Pfleeger, “Principles of Survey Research Part 5: Populations and Samples,” *ACM SIGSOFT Softw. Eng. Notes*, vol. 27, no. 5, p. 17, 2002.

[31] B. a Kitchenham and S. L. Pfleeger, “Principles of Survey Research Part 3: Constructing a Survey Instrument,” *ACM SIGSOFT Softw. Eng. Notes*, vol. 27, no. 2, p. 20, 2002.

[32] B. Kitchenham and S. L. Pfleeger, “Principles of survey research part 4: questionnaire evaluation,” *ACM SIGSOFT Softw. Eng. Notes*, vol. 27, no. 3, p. 20, 2002.

[33] B. Kitchenham and S. L. Pfleeger, “Principles of survey research part 6: Data Analysis,” *ACM SIGSOFT Softw. Eng. Notes*, vol. 28, no. 2, p. 24, 2003.

[34] B. Kitchenham and S. L. Pfleeger, “Principles of Survey Research,” *ACM SIGSOFT Softw. Eng. Notes*, vol. 26, no. 6, p. 16, 2001.

[35] B. a Kitchenham and S. L. Pfleeger, “Principles of Survey Research Part 2 : Designing a Survey Sample size Experimental designs,” *Softw. Eng. Notes*, vol. 27, no. 1, pp. 18–20, 2002.

[36] H. Al-Kilidar, K. Cox, and B. Kitchenham, “The use and usefulness of the ISO/IEC 9126 quality standard,” in *International Symposium on Empirical Software Engineering.*, 2005, pp. 122–128.

[37] R. E. Al-qutaish, “A Maturity Model of Software Product Quality,” *Res. Pract. Inf. Technol.*, vol. 43, no. 4, pp. 307–328, 2010.

[38] E. Van Veenendaal, R. Hendriks, and R. Van Vonderen, “Measuring Software Product Quality,” *SQP*, vol. 5, no. 1, pp. 6–13, 2002.

[39] C. Wohlin, P. Runeson, M. Host, Magnus C. Ohlsson, Bjorn Regnell, and Anders Wesslen, *Experimentation in Software Engineering*. 2012.

[40] P. Torino and P. Torino, “A State-of-the-Practice Survey of Risk Management in Development with Off-the-Shelf A State-of-the-Practice Survey on Risk Management in Development with Off- The-Shelf Software Components,” vol. 34, no. April, pp. 271–286, 2016.

[41] A. Nugroho and M. R. V. Chaudron, “A survey into the rigor of UML use and its perceived impact on quality and productivity,” *Proc. Second ACM-IEEE Int. Symp. Empir. Softw. Eng. Meas. - ESEM ’08*, p. 90, 2008.

[42] J. Soini, “A Survey of Metrics Use in Finnish Software Companies,” *2011 Int. Symp. Empir. Softw. Eng. Meas.*, pp. 49–57, 2011.

[43] W. Chen, J. Li, J. Ma, R. Conradi, J. Ji, and C. Liu, “A survey of software development with open source components in Chinese software industry,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 4470 LNCS, pp. 208–220, 2007.

[44] J. Li, J. Ma, R. Conradi, W. Chen, J. Ji, and C. Liu, “A survey on the business relationship between Chinese outsourcing software suppliers and their outsourcers,” *Proc. - Asia-Pacific Softw. Eng. Conf. APSEC*, pp. 470–477, 2007.

[45] E. Petrinja, A. Sillitti, and G. Succi, “Adoption of OSS Development Practices by the Software Industry: A Survey,” *Open Source Syst. Grounding Res.*, vol. 365, pp. 233–243, 2011.

[46] A. Macphail, T. Hainey, and T. M. Connolly, “Applying Mlearning in Software Engineering Education : a Survey of Mobile Usage,” *Mob. Learn.*, no. iii, 2012.

[47] M. Usman, E. Mendes, and J. Börstler, “Effort estimation in Agile software development: A survey on the state of the practice,” *ACM Int. Conf. Proceeding Ser.*, vol. 27–29–Apri, 2015.

[48] R. N. Memon, S. S. Salim, and R. Ahmad, “Identifying

research gaps in requirements engineering education: An analysis of a conceptual model and survey results," *2012 IEEE Conf. Open Syst.*, pp. 1–6, 2012.

[49] Y. Cerqueira, S. R. De Lemos, Y. C. . Cavalcanti, P. A. . Da Mota Silveira Neto, I. . Do Carmo Machado, E. S. . De Almeida, and S. R. . De Lemos Meira, "Towards Understanding Software Change Request Assignment : a survey with practitioners," *ACM Int. Conf. Proceeding Ser.*, pp. 195–206, 2013.

[50] C. Palomares, C. Quer, and X. Franch, "Requirements reuse and requirement patterns : a state of the practice survey," *Empir. Softw. Eng.*, 2016.

[51] David Ameller, M. Galster, P. Avgeriou, and X. Franch, "A survey on quality attributes in service-based systems," *Softw. Qual. J.*, vol. 24, no. 2, pp. 337–364, 2016.

[52] H. Tsuji, A. Sakurai, K. Yoshida, A. Tiwana, and A. Bush, "Questionnaire-Based Risk Assessment Scheme for," pp. 114–127.

[53] H.-C. Huang, "Freemium business model: construct development and measurement validation," *Internet Res.*, vol. 26, no. 3, pp. 604–625, 2016.

[54] D. Tofan, M. Galster, P. Avgeriou, and D. Weyns, "Software engineering researchers' attitudes on case studies and experiments: An exploratory survey," *Eval. Assess. Softw. Eng. (EASE 2011), 15th Annu. Conf.*, no. 638, pp. 91–95, 2011.

[55] O. Albayrak, "Instructor's Acceptance of Games Utilization in Undergraduate Software Engineering Education: A Pilot Study in Turkey," *2015 IEEE/ACM 4th Int. Work. Games Softw. Eng.*, pp. 43–49, 2015.

[56] D. Budgen, B. A. Kitchenham, S. M. Charters, M. Turner, P. Brereton, and S. G. Linkman, "Presenting software engineering results using structured abstracts: A randomised experiment," *Empir. Softw. Eng.*, vol. 13, no. 4, pp. 435–468, 2008.

[57] F. Q. B. Da Silva and A. C. C. Frana, "Towards understanding the underlying structure of motivational factors for software engineers to guide the definition of motivational programs," *J. Syst. Softw.*, vol. 85, no. 2, pp. 216–226, 2012.

[58] I. Erfurth and W. R. Rossak, "A look at typical difficulties in practical software development from the developer perspective A field study and a first solution proposal with UPEX," *Proc. Int. Symp. Work. Eng. Comput. Based Syst.*, pp. 241–248, 2007.

[59] N. Fenton, M. Neil, W. Marsh, P. Hearty, Ł. Radliński, and P. Krause, "On the effectiveness of early life cycle defect prediction with Bayesian nets," *Empir. Softw. Eng.*, vol. 13, no. 5, pp. 499–537, 2008.

[60] I. Garcia, C. Pacheco, and P. Sumano, "Use of questionnaire-based appraisal to improve the software acquisition process in small and medium enterprises," *Stud. Comput. Intell.*, vol. 150, pp. 15–27, 2008.

[61] J. Hutchinson, J. Whittle, M. Rouncefield, and S. Kristoffersen, "Empirical assessment of MDE in industry," *2011 33rd Int. Conf. Softw. Eng.*, pp. 471–480, 2011.

[62] M. V. Kosti, R. Feldt, and L. Angelis, "Personality, emotional intelligence and work preferences in software engineering: An empirical study," *Inf. Softw. Technol.*, vol. 56, no. 8, pp. 973–990, 2014.

[63] T. Sedano, "Code readability testing, an empirical study," *Proc. - 2016 IEEE 29th Conf. Softw. Eng. Educ. Training, CSEEandT 2016*, pp. 111–117, 2016.

[64] P. . Diebold, A. . Vetró, and D. . Méndez Fernández, "An Exploratory Study on Technology Transfer in Software Engineering," *Int. Symp. Empir. Softw. Eng. Meas.*, vol. 2015–Novem, pp. 86–95, 2015.

[65] A. Forward and T. C. Lethbridge, "Problems and Opportunities for Model-Centric Versus Code-Centric Software Development: A Survey of Software Professionals," *Proc. 2008 Int. Work. onModels Softw. Eng.*, pp. 27–32, 2008.

[66] J. Ji, J. Li, and R. Conradi, "Some lessons learned in conducting software engineering surveys in China," *ESEM'08 Proc. 2008 ACM-IEEE Int. Symp. Empir. Softw. Eng. Meas.*, pp. 168–177, 2008.

[67] J. M. Rojas, G. Fraser, and A. Arcuri, "Automated Unit Test Generation During Software Development: A Controlled Experiment and Think-aloud Observations," pp. 338–349, 2015.

[68] P. Karpati, Y. Redda, A. L. Opdahl, and G. Sindre, "Comparing attack trees and misuse cases in an industrial setting," *Inf. Softw. Technol.*, vol. 56, no. 3, pp. 294–308, 2014.

[69] J. P. Campos, J. L. Braga, A. M. de Resende, and C. H. Os'orio Silva, "Identification of Aspect Candidates by Inspecting Use Cases Descriptions," *SIGSOFT Softw. Eng. Notes*, vol. 35, no. 4, pp. 1–9, 2010.

[70] L. Prechelt and M. Liesenberg, "Design patterns in software maintenance: An experiment replication at Freie Universit??t Berlin," *Proc. - 2011 2nd Int. Work. Replication Empir. Softw. Eng. Res. RESER 2011*, pp. 1–6, 2012.

[71] M. Haddara and A. Elragal, "ERP adoption cost factors identification and classification: a study in SMEs," *Int. J. Inf. Syst. Proj. Manag.*, vol. 1, no. 2, pp. 5–21, 2013.

[72] A. Jedlitschka, "Evaluating a model of software managers' information needs," *Proc. 2010 ACM-IEEE Int. Symp. Empir. Softw. Eng. Meas. - ESEM '10*, p. 1, 2010.

[73] M. Sensalire, P. Ogao, and A. Telea, "Evaluation of software visualization tools: Lessons learned," *2009 5th*

IEEE Int. Work. Vis. Softw. Underst. Anal., pp. 19–26, 2009.

[74] M. Schmidberger and B. Brugge, "Need of Software Engineering Methods for High Performance Computing Applications," *Parallel Distrib. Comput. (ISPDC), 2012 11th Int. Symp.*, pp. 40–46, 2012.

[75] D. Galin, *Software Quality Assurance From theory to implementation*. 2004.

[76] S. H. Kan, *Metrics and Models in Software Quality Engineering*. Addison-Wesley Longman Publishing Co., Inc., 2002.

[77] Rohaiza Abd. Rokis, "Youth Employability and Work Attitudes," *Int. J. Sci. Commer. Humanit.*, vol. 2, no. 5, p. 13, 2014.

[78] G. Vigderhous, *The Level of Measurement and Permissible Statistical Analysis in Social Research*, vol. 20, no. 1. 1977.

[79] U. Jakobsson, "Statistical presentation and analysis of ordinal data in nursing research," *Scand. J. Caring Sci.*, vol. 18, no. 4, pp. 437–440, 2004.

[80] IEEE Computer Society, "IEEE Standard for a Software Quality Metrics Methodology - IEEE Std 1061™-1998 (R2009)," vol. 1998, 2009.

[81] H. Jung and S. Kim, "Measuring Software Product Quality: A Survey of ISO/IEC 9126," *IEEE Softw.*, pp. 88–92, 2004.

[82] L. M. Lozano, E. García-Cueto, and J. Muñiz, "Effect of the Number of Response Categories on the Reliability and Validity of Rating Scales," *Methodol. Eur. J. Res. Methods Behav. Soc. Sci.*, vol. 4, no. 2, pp. 73–79, Jan. 2008.

[83] C. Alzola and F. Harrell, "An Introduction to S and the Hmisc and Design Libraries," 2006. [Online]. Available: http://her.gr.distfiles.macports.org/mirrors/CRAN/doc/contrib/Alzola+Harrell-Hmisc-Design-Intro.pdf. [Accessed: 29-Jan-2015].

[84] "Technology Evaluation Centers." [Online]. Available: http://www.technologyevaluation.com/. [Accessed: 18-Apr-2017].

[85] J. Baroudi, M. Olson, and B. Ives, "An empirical study of the impact of user involvement on system usage and information satisfaction," *Commun. ACM*, vol. 29, no. 3, pp. 232–238, 1986.

[86] Software Engineering Institute, "CMMI® for Development, Version 1.3," 2010.

[87] IEEE Computer Society, *SWEBOK Guide*. 2014.

[88] S. Hansen and J. Rennecker, "Getting on the same page: Collective hermeneutics in a systems development team," *Inf. Organ.*, vol. 20, no. 1, pp. 44–63, Jan. 2010.

[89] O. Braud, "Facteurs decisionnels pour l'iplantation d'un ERP dans les PME : Le role de l'evaluation des benefices tangibles et intangibles," 2008.

[90] A. J. Albrecht, "Measuring application development productivity," *IBO Conf. Appl. Dev.*, vol. 10, pp. 83–92, 1979.

[91] R. Alvaro, "Framework for a global quality evaluation of a website," *Online Inf. Rev.*, vol. 36, no. 3, pp. 347–382, 2012.

[92] H. Yang, "Measuring software product quality with ISO standards base on fuzzy logic technique," *Adv. Intell. Soft Comput.*, vol. 137 AISC, pp. 59–67, 2012.

[93] P. Tomas, M. J. Escalona, and M. Mejias, "Open source tools for measuring the Internal Quality of Java software products. A survey," *Comput. Stand. Interfaces*, vol. 36, no. 1, pp. 244–255, 2013.

[94] M. Kwiatkowski and C. Verhoef, "Recovering management information from source code," *Sci. Comput. Program.*, vol. 78, no. 9, pp. 1368–1406, 2013.

[95] S. Lehtonen, "Metrics for Gerrit code reviews," no. May, pp. 31–45, 2015.

[96] B. H. Layne, J. R. Decristoforo, and D. Mcginty, "Electronic versus traditional student ratings of instruction," *Res. High. Educ.*, vol. 40, no. 2, pp. 221–232, 1999.

[97] B. Kitchenham, O. Pearl Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, "Systematic literature reviews in software engineering - A systematic literature review," *Inf. Softw. Technol.*, vol. 51, no. 1, pp. 7–15, 2009.

[98] Ameller David, Galster Matthias, Paris Avgeriou, and Franch Xavier, "A survey on quality attributes in service-based systems," *Softw. Qual J*, 2016.

[99] K. Dullemond and B. van Gameren, "What Distributed Software Teams Need to Know and When: An Empirical Study," *Glob. Softw. Eng. (ICGSE), 2013 IEEE 8th Int. Conf.*, pp. 61–70, 2013.

[100] J. M. Rojas, G. Fraser, and A. Arcuri, "Automated unit test generation during software development: a controlled experiment and think-aloud observations," *Proc. 2015 Int. Symp. Softw. Test. Anal. - ISSTA 2015*, pp. 338–349, 2015.

[101] A. Jedlitschka, "Evaluating a model of software managers' information needs," *Proc. 2010 ACM-IEEE Int. Symp. Empir. Softw. Eng. Meas. - ESEM '10*, p. 1, 2010.

[102] A. Nugroho and C. F. J. Lange, "On the relation between class-count and modeling effort," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 5002 LNCS, pp. 93–104, 2008.

APPENDIX C – ARTICLE 3: DOES PARTICIPANTS' KNOWLEDGE AFFECT THE SURVEY RESULTS? A SYSTEMATIC LITERATURE REVIEW IN SOFTWARE ENGINEERING DOMAIN

Reza Mirsalari
Laboratoire de génie logiciel
École Polytechnique de Montréal
Montréal, Canada
reza.mirsalari@polymtl.ca

Pierre N. Robillard
Laboratoire de génie logiciel
École Polytechnique de Montréal
Montréal, Canada
Pierre-n.robillard@polymtl.ca

Submitted to AST journal

ABSTRACT

Context: Along with the growth of the studies in software engineering domain, questionnaire-based surveys have also been of interest to researchers. There are many factors influencing the results of the questionnaires, such as questions wording, questionnaire formatting, the respondents, the generalizability of the results, etc. **Objective**: In this research, we focused on the effects of the participants' knowledge on the survey's results since it is rarely addressed in the literature, while it can affect the validity of the survey. **Method**: we performed a systematic literature review to reveal that how the researchers in software engineering are conducting the surveys and what factors they are neglecting. In addition, we conducted two surveys in a software consulting company to check if the participants' knowledge affects the survey results. **Results**: Our systematic literature review shows that the existence of this effect is controversial among the researchers. We also presented the results of our two surveys in this paper. **Conclusion**: Our surveys approve that the participants' knowledge may affect the result of the surveys. The audience of this paper is the researchers who will manage a survey and/or evaluate survey results.

KEYWORDS

Software Engineering, Systematic Literature Review, Survey, Questionnaire, Software quality, Participants' knowledge

1. INTRODUCTION

Software engineers use surveys to highlight complex issues, to provide a solution and support effective decision making. There is a subtle distinction between the terms Survey and Questionnaire. The survey is a comprehensive process of data collection and analysis while the questionnaire is the instrument used during the survey process. This instrument can be of many types such as paper questionnaire, telephone interview, etc. Kitchenham et al. advised that for conducting a survey, at the first step, the objectives, the schedule, and the resources should be determined. In the next steps the survey should be designed, the instrument should be selected, the survey should be run, and at the end, the researcher will be able to analyze the data and create the final report [34][39]**.**

Conducting a survey can be used in several ways. The traditional and typical ones such as mail surveys, street-surveys, and telephone surveys are those that use simple data collection medium and most often result in low response rate. The evaluations are paper-and-pencil based and time-consuming [96]. However, the internet made the web-based surveys possible. The advantages of the web-based – or electronic – surveys are the possibility to collect a large amount of data with low cost and without manual data entry from paper questionnaire into a digital file. The electronic surveys are of two types, email surveys, and online surveys. In the email surveys, the participants receive an email with the questionnaire attached to it. They download the attached questionnaire, open it in another application – or print it, fill it in, and send it back to the researcher. In the online surveys, the procedure is easier. The invited participants are asked to open a web page, answer the questions and click on Submit button at the end. Although each survey type has advantages and disadvantages, the electronic surveys are more efficient and popular nowadays [28]. However, there are factors that often influence the surveys, regardless of what kind of instrument and method has been applied.

1.1. Data Collection Methods in Software Engineering

Research in software engineering field is deeply involved with “people”. People – or in particularly ‘the stakeholders’ – including customers, developers, end users, and maintainers are systematically intervened in the software lifecycle. Consequently, field studies in software engineering are significant if the researchers conduct the study using the “*real people in the real environment*” [29].

However, collecting data from the stakeholders should be carefully taken into consideration. From all the available techniques, researchers should select the most appropriate ones to collect the most reliable information to obtain the most accurate results.

Lethbridge et al. presented a taxonomy of data collection techniques in [29]. This taxonomy is based on the degree of human contact each data collection technique requires. Depending on the nature of the research, other techniques such as brainstorming, shadowing, checklist, or a combination of techniques can be applied. From the 14 different techniques presented in [29], *interview* and *questionnaire* are the most popular ones in software engineering field studies [29]. How are the questionnaires defined in software engineering?

1.2. Questionnaires in Software Engineering

By definition, 'questionnaire' is a set of written questions formed for addressing a specific research question. There are many factors influencing the results of the questionnaires, such as questions wording, questionnaire formatting, the respondents, the generalizability of the results, etc. Pfleeger and Kitchenham have presented a six-part series on principles of survey research. They explained – in detail – how to design, construct, and conduct a survey in addition to the threats and pitfalls in survey administration and data analysis.

Questionnaire-based surveys are becoming popular in software engineering community. According to our simple search in Inspec digital library, in 2007 only 37% of the studies in software engineering have addressed a questionnaire-based survey, while this increased to 63% in 2016, i.e. 70% growth (Figure 32). One of the reasons for this increase can be the features that electronic-based survey tools provide for the researchers. Nowadays, researchers create their electronic and web-based questionnaires efficiently with friendly graphical user interfaces. This makes the creation, global submission, data collection, and results analysis simpler than before when the paper-based questionnaires were being used.

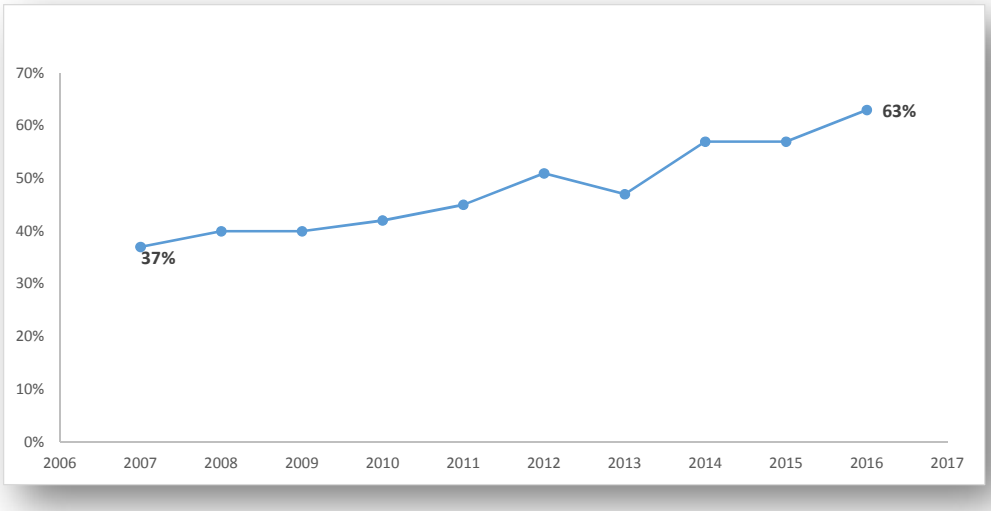


Figure 32: Growth of questionnaire-based surveys in SE since 2007

To understand how the researchers perform the surveys in SE domain, and what factors have been considered or neglected, a systematic literature review is conducted and the results are presented in this paper

1.3. Systematic Literature Review in Software Engineering

We conducted a systematic literature review (SLR) to extract the papers that are related to the research topic. In the next sections, the topic, the methodology and the findings of our SLR will be discussed. Firstly, we introduce a systematic literature review as "*a means of evaluating and interpreting all available research relevant to a particular research question or topic area or phenomenon of interest*" [97]. In SLR, the researcher collects and summarizes the primary papers and then attempts to offering new general insights or tackle a specific research question. Since conduction of the surveys is complex human-related projects, many factors should be taken into consideration with care, such as "population selection", i.e. the size and the quality of the participants in the surveys.

1.4. "Participants' knowledge" in Software Engineering surveys

Defining an appropriate target population is essential in designing a survey. The target population is the group of representative audiences that have the relevant knowledge to participate and answer the survey's questions. Selecting the correct population is important because this correctness

affects the validity of the survey results, i.e. the more the population is selected with care, the more the survey results can be generalizable [30].

Unfortunately, the role of participants' knowledge is neglected in the literature. There are few reports that highlight the importance of hiring knowledgeable participants or discussing the threats that selecting unknowledgeable participants may influence the results of the survey. Thanks to web-based questionnaires, many researchers simply attempt to increase the number of participants as much as they can, with no care about the risks that the answers from the unknowledgeable participants might cause for the validity of the results. In this paper, we raise the participant's knowledge concern among the published reports in the literature and try to summarize the ones that address this concern.

In this study, firstly we formulate a research question, then we develop the SLR protocol. For conducting the SLR, we identify the relevant reports, then we assess the quality of the papers and extract the required data. We end up our SLR by synthesis the results and packaging the project. For demonstrating the syntheses of the SLR, we conducted two surveys in a software selection company. The results and the analyses are also included in this paper.

We begin in the next section by providing an overview of the systematic literature review and describing how the relevant papers are retrieved and how the synthesis is created. In Section 3, we talk about each of the two surveys that we conducted in a software consulting company. The results and the discussions are also presented in the subsections. We talk briefly in Section 4 about items that threat the validity of our study. The paper is concluded in Section 5.

2. SYSTEMATIC LITERATURE REVIEW

In this SLR, it is aimed to investigate on how the researchers in software engineering conduct the surveys and how they report the general issues. Specifically, it is also targeted to investigate whether the "participants' technical knowledge" has been addressed as a concern when researchers conduct a survey.

2.1. Data gathering for Systematic Literature Review

The SLR was performed in two styles: a *systematic* literature review and a *wild* literature review. In the systematic style, we defined the sources, created a comprehensive search string and then we retrieved, refined, and analyzed the data. In the wild style, we did a simple search, generally on

Google scholar, for the specific author(s) or specific paper(s) that we guessed have some related information. Table 49 shows the search string that was used to retrieve the related papers.

Table 49: Search string for SLR

Keywords	include	"software engineering"
		AND
Title	include	"survey" OR "questionnaire"
		AND
Keywords	include	"problem*" OR "difficulty*" OR "issue*" OR "participant's knowledge" OR "knowledge of participants" OR "population's knowledge" OR "knowledge of population"
		AND
Document type	equals to	conference article OR journal article
		AND
Publication year	between	2007—2017
		AND
Language	equals	English only

As it can be seen, we limited our search to English conference or journal papers that were publish since the year of 2007. The retrieved papers were extracted from the electronic libraries, where the high-quality papers in SE are published. The targeted electronic libraries are:

- IEEE-Xplore
- Compendex/Inspec
- ACM
- Web of Science
- Wild search: Google scholar

Applying the search string to the libraries resulted in retrieving 242 records. In the next step, we excluded the papers with the following conditions:

- duplicate papers
- less than 5 pages
- title not related
- abstract not related
- full text not related
- full text (pdf) unavailable

Finally, 34 related papers were retrieved. Table 50 shows the details of the selection process.

Table 50: Details of paper selection and elimination in the SLR

	Compendex/Inspec	ACM	IEEE Xplore	WEB of Science	Wild (Google Scholar)	**Total**
Total retrieved	139	8	30	36	14	**227**
Duplicates	28	1	5	1	0	**33**
Pages<5	10	2	9	4	5	**30**
Title not-related	33	0	0	0	0	**33**
Abstract or body not-related	56	0	12	18	3	**87**
PDF unavailable	0	0	1	5	0	**6**
Related	**12**	**5**	**3**	**8**	**6**	**34**

From the retrieved papers the following context values were extracted:

- Research question or the message of the paper: to make sure that the retrieved paper is related to our SLR topic
- What type of validity is performed? To retrieve the reported issues, since the researchers may address the issues when they are reporting the validity checks.
- What is the population size of the conducted survey? To check if there is a relationship between the population size and the other context values.
- Is the survey pre-tested before the conduction? Kitchenham et al. advised that it is useful to pre-test the survey before the official conduction. Have the researchers pre-tested the surveys?
- Have the participants been selected with care considering their related technical knowledge?
- Is the effect of participants' knowledge on the survey results addressed?

2.2. Retrieved papers

In this section, a summary of the 34 retrieved papers is presented. First, the aim or the research question of each paper is presented. Then in Table 51, the following context values are retrieved.

- *Paper ID: first Author, year*
- *Population: the number of the participants in the survey*
- *Validation: Whether the survey is validated in terms of internal validation, external validation, etc.*
- *Pre-test: Whether the questionnaire is pre-tested or pre-piloted*
- *Selection: Whether the participants are selected with care*
- *Impact: Whether the effect of participants knowledge is addressed*

Here is the report:

1. **Li_2008** [40]: The impacts of Off-the-shelf components on software quality. The 20-question questionnaire in this project uses the five-point Likert scales to collect background information about the participant company and the respondents. Questions are especially on risk management and process improvement. The researchers used stratified random sampling to select companies and convenience sampling to select projects inside a company.
2. **Nugroho_2008_1** [41]: The 20-question questionnaire is aimed to collect data and investigate the opinions from professional software engineers about different styles used in UML modeling and their perceived impacts on quality and productivity in software development.
3. **Soini_2011** [42]**:** "This paper uses the results of an empirical case study to examine how the measurement was implemented in practice from the perspective of the software process". The 13-questions questionnaire aimed to collect separate metrics of customer feedback information.
4. **Chen_2007** [43]**:** This paper addressed the issues in reusing open-source components. The questionnaire aims to collect information "on three main issues in reusing OSS components for software development in Chinese software industry, namely component selection, licensing terms, and system maintenance."
5. **Li_2007** [44]**:** This paper reviews the relationship between the Chinese suppliers and outsourcers. The questionnaire aims to collect data and "investigate how Chinese suppliers have built and maintained partnership or contract relationship with their outsourcers, and the effect of these relationships on the success of outsourcing".
6. **Petrinja_2011** [45]**:** To "collect data related to practices and elements in the development process of companies that influence the trust in the quality of the product by potential adopters".
7. **Macphail_2012** [46]**:** To show how mLearning is suitable for teaching.

8. **Usman_2015** [47]**:** To report on the state of the practice on effort estimation in agile software development. The questionnaire aims to collect data on effort estimation techniques from agile teams.
9. **Memon_2012** [48]**:** In this paper, a questionnaire was created to investigate lecturers' perceptions of the Requirement Engineering Education problems presented in integrated viewpoint. "The data collection was aimed at lecturers who have taught RE course."
10. **Cavalcanti_2013** [49]**:** This paper reviews the impact of Change Request assignment on software development.
11. **Palomares_2016** [50]**:** This paper is "an exploratory study of the practices in requirements reuse that are currently being used in organizations and to study in more depth the possible benefits and drawbacks of the use of patterns as a requirement reuse technique".
12. **Ameller_2016** [98]**:** This paper aims to review the quality attributes in service-based systems. The questionnaire collects the data about the significance of quality attributes when designing service-based systems and how quality attributes are addressed through design decisions.
13. **Tsuji_2007** [52]**:** The research question is formulated as "Although a lot of engineers have experienced the success and failure of their projects, their know-how still remains as tacit knowledge. This paper proposes a risk assessment scheme for new projects by externalizing such tacit knowledge".
14. **Hao_2015** [53]**:** "The purpose of this paper is to probe into the development of the dimensions of the freemium business model and validate the measurement".
15. **Tofan_2011** [54]**:** This paper aims to answer that "How do empirical software engineering researchers perceive the differences between case studies and experiments, and how do perceptions of researchers vary along their views on the nature of case study".
16. **Albayrak_2015** [55]**:** this paper addresses the impact of three factors on the instructor's decision to use games on in software engineering education. The factors are 1) the number of hours per week the instructor plays game 2) instructor's experience in using games for educational purposes in general, and 3) instructor's experience in designing games
17. **daSilva_2012** [57]**:** This paper attempts to present a better understanding of 'motivation' in software engineering.
18. **Erfurth_2007** [58]**:** This paper "presents a look at typical difficulties in practical software development from the developer perspective". The questionnaire aims get answers regarding the state of art in practice.
19. **Fenton_2007** [59]**:** This paper discusses an experiment to develop a causal model (Bayesian net) for predicting the number of residual defects that are likely to be found during independent testing or operational usage.
20. **Garcia_2008** [60]**:** To show the application of a "Maturity Questionnaire" in a disciplined way. The proposed questionnaire focuses in Supplier Agreement Management Process Area of the CMMI.
21. **Kosti_2014** [62]**:** This paper tries to show that "the associations can help managers to predict and adapt projects and tasks to available staff". The results also show that the Emotional Intelligence instrument can be predictive.

22. **Sedano_2016** [63]**:** This paper tries to show that Code Readability Testing improves programmers' ability to write readable codes. A questionnaire is created to retrieve the programmers' perspective.
23. **Dullemond_2011** [99]**:** This paper investigates on Conversation as a communication pattern. It explores the importance of overhearing conversations by conducting a questionnaire in a large international software development company.
24. **Forward_2008** [65]**:** The survey presented in this survey aims to extract the participants' attitudes and experiences regarding software modeling, and development approaches that avoid modeling.
25. **Rojas_2015** [100]**:** The authors discussed the knowledge of the participants: "Participants without sufficient knowledge of Java and JUnit may affect the results; therefore, we only accepted participants with past experience".
26. **Karpati_2014** [68]**:** The 35 participants were randomly assigned to one of four experiment groups. The participants in groups 1 and 2 used Misuse Cases before Attack Trees, whereas groups 3 and 4 used them in the opposite order.
27. **Campos_2010** [69]**:** Two case studies on two student groups with different level of knowledge were conducted in this study. The first case study was carried out with 19 voluntary students of the Software Engineering II discipline. The second was carried out with 7 voluntary students that were taking the Software Engineering discipline as part of their Master's Degree program in Computer Science. The dependent variable was measured before and after a training process.
28. **Prechelt_2011** [70]**:** In this study, it is reported that an experiment was conducted then a short course was offered to the participants to increase their knowledge of design patterns. The researchers asked for prior knowledge of 17 particular design patterns on a scale from 1, 2, or 3 (described as "never heard of it", "have only heard of it", and "understand it roughly", respectively) up to 7 ("understand it well and have worked with it many times").
29. **Haddara_2013** [71]**:** The collected data was based on the participants' knowledge and experience from completed ERP projects in SMEs. "This research explores the direct and indirect cost factors that occur in ERP adoptions in Egyptian SMEs".
30. **Jedlitschka_2010** [101]**:** In this study, the researchers aimed to evaluate the effectiveness of the model of software managers information needs. The researchers asked respondents regarding their involvement in the decision-making process, their general business experience in years as well as their experience with technology selection, and their knowledge regarding the technologies. The questionnaire collects the information about the baseline experiment.
31. **Sensalire_2009** [73]**:** "This paper presents the lessons learned from evaluating over 20 tools with over 90 users in five different studies spread over a period of over two years.
32. **Nugroho_2008_2** [102]**:** this paper investigates the usefulness of class-count as a size measure of UML models. By using two student experiments the researchers validated this measure by assessing its correlation with effort spent in modeling.
33. **Budgen_2008** [56]**:** The paper shows "whether structured abstracts are more complete and easier to understand than non-structured abstracts for papers that describe software engineering experiments. […] The participants were each presented with one abstract in its

original unstructured form and one in a structured form, and for each one were asked to assess its clarity (measured on a scale of 1 to 10) and completeness (measured with a questionnaire that used 18 items).

34. **Ji_2008** [66]**:** This paper addressed the issues and the lessons learned from conducting a survey in china relating to sampling, contacting respondents, data collection, and validation. The paper presents a lesson as "[…] people with sufficient software engineering knowledge need to be involved in the data collection process in order to respond to any confusion on the part of the respondents, and to ensure that the qualifications of the respondents are appropriate […] "Many project managers and developers do not have sufficient knowledge of software engineering methods and terms. It is necessary to supervise the entire data collection process, such as training people to assist in the survey, answering questions from respondents, and validating the quality of completed questionnaires".

Table 51 : Descriptions: Paper's context values

	Paper ID	Population	Validation	Pre-Test	Selection	Impact
1	Li_2008	133	Internal, External, Construct, Conclusion	Yes	NR	NR
2	Nugroho_2008_1	80	Internal, External	NR	Yes	NR
3	Soini_2011	40	NR	NR	NR	NR
4	Chen_2007	47	Construct, Internal, External	Yes	NR	NR
5	Li_2007	53	Construct, Internal, External	NR	NR	NR
6	Petrinja_2011	56	NR	NR	NR	NR
7	Macphail_2012	321	NR	NR	NR	NR
8	Usman_2015	63	Construct, Internal, External, Conclusion	Yes	No	No
9	Memon_2012	18	NR	NR	NR	NR
10	Cavalcanti_2013	36	NR	Yes	Yes	NR
11	Palomares_2016	71	Internal, External, Construct	Yes	Yes	NR
12	Ameller_2016	56	External, Randomization, Exclusion	Yes	Yes	NR
13	Tsuji_2007	175	NR	NR	NR	NR
14	Hao_2015	1061	NR	Yes	Yes	NR

Table 51 : Descriptions: Paper's context values

	Paper ID	Population	Validation	Pre-Test	Selection	Impact
15	Tofan_2011	26	External, Internal, Construct	Yes	NR	NR
16	Albayrak_2015	30	NR	NR	NR	NR
17	daSilva_2012	176	External, Construct	NR	NR	NR
18	Erfurth_2007	65	NR	NR	NR	NR
19	Fenton_2007	31	External	NR	NR	NR
20	Garcia_2008	600	NR	NR	NR	NR
21	Kosti_2014	272	Conclusion	NR	NR	NR
22	Sedano_2016	21	Construct, Internal, External	NR	NR	NR
23	Dullemond_2011	44	Conclusion	NR	NR	NR
24	Forward_2008	113	NR	NR	NR	NR
25	Rojas_2015	41	NR	Yes	NR	NR
26	Karpati_2014	35	Construct, Conclusion, Internal, External	Yes	NR	NR
27	Campos_2010	26	NR	NR	Yes	Yes
28	Prechelt_2011	41	NR	Yes	NR	NR
29	Haddara_2013	NR	NR	NR	Yes	NR
30	Jedlitschka_2010	20	Construct, Conclusion, Internal, External	Yes	Yes	Yes
31	Sensalire_2009	90	NR	NR	Yes	Yes
32	Nugroho_2008_2	12	NR	NR	Yes	Yes
33	Budgen_2008	64	Construct, Internal	NR	NR	NR
34	Ji_2008	NR	NR	Yes	NR	NR

2.3. Findings

2.3.1. The leading research

The leading research that we have found among our SLR was the six-part series on principles of surveys research created by Pfleeger and Kitchenham [34][35][31][32][30][33]. The series presents a comprehensive and detailed knowledge about questionnaire design and conduction in software engineering. However, other researchers, such as Punter et al. continued this endeavor by introducing the essentials on conducting the web-based questionnaires in software engineering. According to all the reviewed studies, the questionnaires are all implemented in web-based style.

2.3.2. Population size

In order to attain a set of valuable and reliable responses from a survey, it is crucial to determine the ideal sample size and population for the survey [30]. One of the items that we investigated in our literature review was the population size of the surveys. We clustered the results into three groups:

- Less than 100 participant surveys: in average 43 participants participated in each survey
- Between 100 to 500 participant surveys: in average 215 participants per survey
- More than 500 participant surveys: in this category we retrieved only two surveys; one with 1061 [53] and another with 600 [60] participants.

2.3.3. The lacks among the papers

Our results of the SLR revealed some lacks in the studies. The lacks are the items that have been advised by the leading research, but they are not met during the surveys conduction. These missing items include ‘lacking the pre-test’, and ‘lacking the participant-selection’ processes.

- **Pre-test:** Although Kitchenham advises that the questionnaire should be pre-tested (or pre-piloted) to check if the questions are understandable, and the whole questionnaire is reliable and valid [32], many of our retrieved papers (21 out of 34) have not reported if they have evaluated their surveys through a pre-test procedure. The lack of pre-test evaluation might result in submitting a set of vague questions to the participants and introducing unreliable and invalid survey instrument. It may also cause to select the data analysis technique that doesn’t match the expected answers.
- **Participant selection:** Kitchenham advises that before the survey is conducted, the target and the sample population should be determined. It is also advised strongly that the sample population should be representative [30]. The results of our SLR shows that in 21 out of

34 studies (almost 50%) the importance of participants and the role of participant selection process is not reported.

2.3.4. Participants' knowledge

One of the findings that strongly affects the quality of the surveys in software engineering is the technical knowledge of the participants in the survey. The participants' knowledge is related to and should be considered during the procedure of participant selection. This item has also been highlighted by few researchers while mostly ignored by others. In this section, we divide the retrieved papers into three categories: the positives, the negatives, and the neutrals. The number of papers in each category shows that the concern of the participant's knowledge is reported in 5 papers, while it is ignored in 29 others. Among the 5 papers, 3 papers claimed that the effect exists, and 2 papers showed that participants' knowledge doesn't affect the survey results.

- The Positives – 3 papers

Namely, the Positive papers are those that claim that the participants' knowledge does affect the results of the survey. The Positive papers are presented in Table 52.

Table 52: The papers that report that participants' knowledge AFFECTS the survey results

Paper code	Population size	Validity	Pre-Test	Participants picked with care
Rojas_2015	41	Not reported	Yes	Yes
Sensalire_2009	90	Not reported	Not reported	Yes
Nugroho_2008_2	12	Not reported	Not reported	Yes

- The Negatives – 2 papers

The papers that claim that the participants' knowledge doesn't affect the result of the survey. These papers are presented in Table 53.

Table 53: The papers that report that there is NO effect of participants' knowledge on survey results

Paper code	Population size	Validity	Pre-Test	Participants picked with care
Campos_2010	26	Not reported	Not reported	Yes
Jedlitschka_2010	20	Checked	Yes	Yes

- The Neutrals – 29 papers

The papers that address the participants' selection among conducting the survey, but they didn't check if that affect the survey results. In addition, the papers that didn't address the participants' knowledge at all, are also included in this category. The papers are presented in Table 54.

Table 54 : The papers that do not care (NEUTRAL) about the effect of participants' knowledge on survey results

Paper code	Population size	Validity?	Pre-Test?	Participants were Picked with Care?
Li_2008	133	Checked	Yes	Yes
Nugroho_2008_1	80	Checked	Not reported	Yes
Soini_2011	40	Not reported	Not reported	Not reported
Chen_2007	47	Checked	Yes	Not reported
Li_2007	53	Checked	Not reported	Not reported
Petrinja_2011	56	Not reported	Not reported	Not reported
Macphail_2012	321	Not reported	Not reported	Not reported
Usman_2015	63	Checked	Yes	Not reported
Memon_2012	18	Not reported	Not reported	Not reported
Cavalcanti_2013	36	Not reported	Yes	Yes
Palomares_2016	71	Checked	Yes	Yes
Tsuji_2007	175	Not reported	Not reported	Not reported
Hao_2016	1061	Not reported	Yes	Yes
Tofan_2011	26	Checked	Yes	Not reported
Albayrak_2015	30	Not reported	Not reported	Not reported

Paper code	Population size	Validity?	Pre-Test?	Participants were Picked with Care?
daSilva_2012	176	Checked	Not reported	Not reported
Erfurth_2007	65	Not reported	Not reported	Not reported
Fenton_2007	31	Checked	Not reported	Not reported
Garcia_2008	600	Not reported	Not reported	Not reported
Kosti_2014	272	Checked	Not reported	Not reported
Sedano_2016	21	Checked	Not reported	Not reported
Dullemond_2011	44	Checked	Not reported	Not reported
Forward_2008	113	Not reported	Not reported	Not reported
Karpati_2014	35	Checked	Yes	Not reported
Prechelt_2011	17	Checked	Yes	Yes
Haddara_2013	NR	Not reported	Not reported	Not reported
Budgen_2008	64	Checked	Not reported	Not reported
Ji_2008	NR	Not reported	Yes	Not reported
Ameller_2016	56	Checked	Yes	Yes

2.4. Conclusion of the SLR

In this research, we investigated the effects of the participants' knowledge on the questionnaire's results since it is rarely addressed in the literature. Our systematic literature review shows that the

existence of this effect is *controversial* in the researchers' viewpoint. From our 34 papers set, 3 papers approve the existence of the effect, 2 papers claimed that knowledge of the participants has no effect and the others (29 papers) either did not address the importance of participants' knowledge, or they paid attention to participants' knowledge but they have not checked if there is a cause/effect impact on the results of the survey.

3. CASE STUDY: CONDUCTING TWO SURVEYS

To demonstrate whether the participants' knowledge affects the survey results, we conducted two surveys in a software consulting company named TEC[2] to evaluate the quality profile of the software they are using. In Figure 33 the whole process for creating the quality profile is explained. The research process is composed of three phases: theoretical research, conduction the experiments, and analysis of the results. The research phase begins with creating the questionnaire, then questions are refined through a Delphi method and the answers of the participants are analyzed. The research ends up with creating the quality profile.

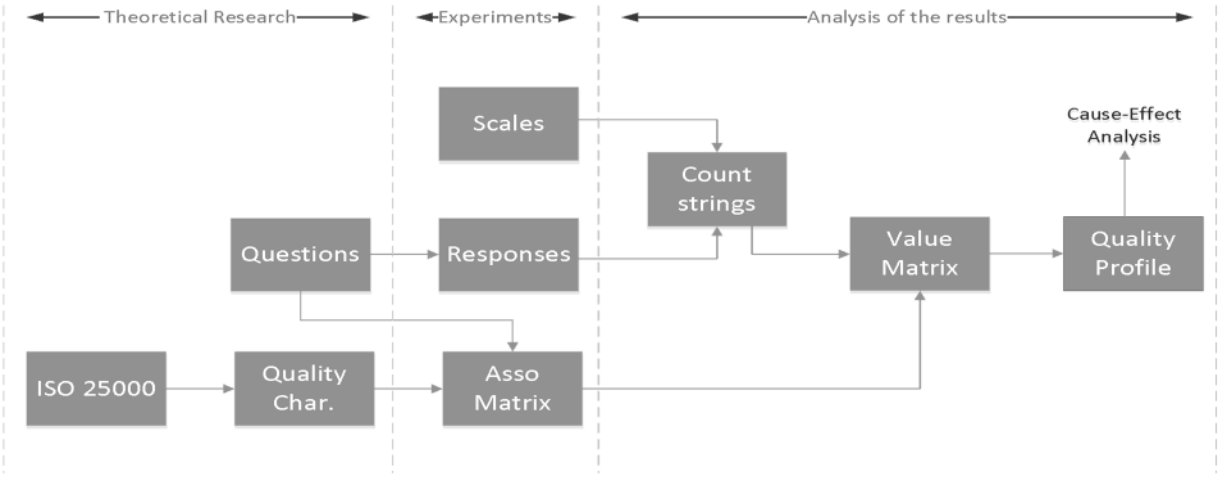


Figure 33: Research process to create the Quality Profile

[2] *"TEC helps other companies to investigate, evaluate, and select the best enterprise software solutions for their unique business requirements. From small businesses to large enterprises, its clients include hundreds of private- and public-sector organizations in a variety of industries. A complete list of vendors and products serves the TEC software selection methodology supporting the software acquirers to make an efficient decision"* [84].

Since we have already published the details of the questionnaire creation and refinement procedure in previous papers [1][9], here we only focus on the results of the surveys to investigate if the participants' knowledge affects the results of the survey. Suffice to say, the questionnaire aims to create the product quality profile. Creating the profile includes eliciting the users' quality requirements, and then quantifying the elicited quality factors by applying them to the quality evaluation model and the ISO/IEC 25000 series of standards. The subject of the questionnaire can be any software product, which is deployed at any organization.

The objective of the surveys was to investigate the effects of participants' knowledge on the results of the questionnaires. The purpose was to evaluate the quality of Microsoft SharePoint (SP) at TEC. The questionnaire includes 33 questions. In the 1st questionnaire, we asked the employees of *Software Development department* to participate. We call the participants of the 1st questionnaire as "tech-savvy" participants who are the employees familiar with software development technologies, programming languages, user interfaces, and common functions and feature of the most popular software applications. They are technically literate and spend almost all their daily time working with software applications. 33 employees were asked to participate, 17 employees responded (contribution rate = 50%)

In the 2nd questionnaire, we asked the rest of the TEC's employees to participate (all the employees but the developers). We call the participants in the 2nd questionnaire as non-technical participants whose daily job doesn't force them necessarily to work with almost any professional software applications. They are technically novices who occasionally work with general software applications. They spend daily a few minutes working with a software application. 110 employees were asked to participate, 30 of them do not use SharePoint at all in their daily jobs. 27 (out of 80) employees responded the questionnaire (contribution rate = 34%).

For each question, an identical set of Likert scales were provided. The scales were coded from A to E as in Table 55.

Table 55 : The Likert scales

Code	Likert Scale
A	Fully supported
B	Mostly supported
C	Fairly supported
D	Poorly supported
E	Not supported
X	Don't know

3.1. The Results

The whole data show that the respondents in the 2nd questionnaire have selected more Don't Know (X) options, than the 1st questionnaire respondents. In other words, the non-technical participants (in the 2nd questionnaire) have less knowledge about the SharePoint quality factors at TEC than the technical users (in the 1st questionnaire). The chart in Figure 34 shows the results in more details. These data are normalized.

During the data analysis process of the questionnaire, the answers were associated with the standard quality characteristics. Then they were categorized in three scales; 'supported', 'so-so', and 'not supported', as follow:

$$Supported = A + B = [Fully\ Supported] + [Mostly\ Supported] \quad (1)$$

$$So\ So = C = [Fairly\ Supported] \quad (2)$$

$$Not\ Supported = D + E = [Poorly\ Supported] + [Not\ Supported] \quad (3)$$

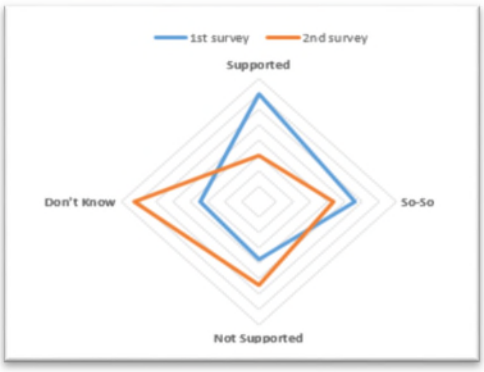


Figure 34: The scales in both surveys

The charts in Figure 35, Figure 36, Figure 37, and Figure 38 show the details of the breakdown distribution of the resulted quality characteristics. These charts illustrate that the respondents in the 1st questionnaire (who are the technical users) believe that SharePoint at TEC supports more quality characteristics than the respondents in the 2nd questionnaire (who are non-technical users). To have a set of valid data for comparison, all the comparative data have been normalized. Table 56 shows a real example of data before and after the normalization process. The raw data are retrieved directly from counting the answers and then they have been normalized to 0-100 range. For example, according to Table 56, for Effectiveness, we have received 12 'Fully Supported' answers. It means that 4% of all the answers received for all the scales for Effectiveness belong to 'Fully Supported'. The same argument shows that 42% of the answers belong to 'Don't know'.

Table 56: The scales and their normalized values for Effectiveness in the 2nd questionnaire

Effectiveness		
Scale	*Raw*	*Normalized*
Fully Supported	12	4%
Mostly Supported	19	7%
Fairly Supported	62	22%
Poorly Supported	53	19%
Not Supported	16	6%
Don't know	116	42%
TOTAL	**278**	**100%**

Consequently, the vertical axes in the following radar charts are in 0-100 range. For example, the chart titled '*Don't Know – Product Quality*' shows that 30% of the Don't Know answers from the 1st questionnaire is related to Functional Suitability, while this value is 40% in the 2nd questionnaire.

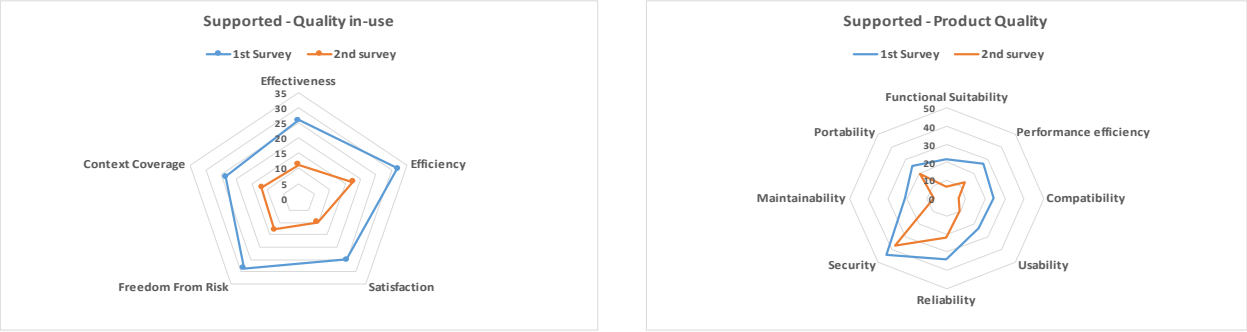


Figure 35: Quality Profiles for 'Supported' scale

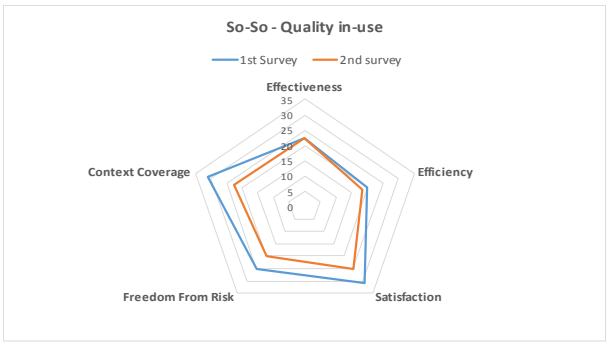


Figure 36: Quality Profiles for ‘So-So’ scale

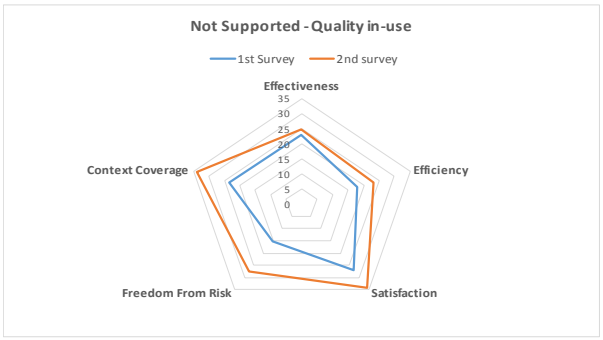
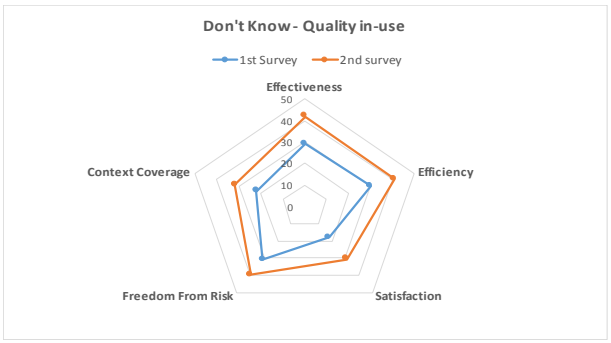


Figure 37: Quality Profiles for ‘Not Supported’ scale



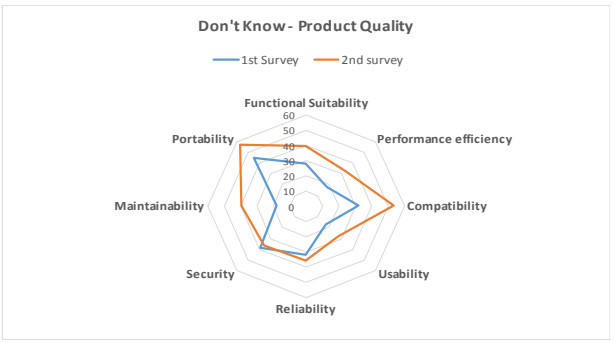


Figure 38: Quality Profiles for ‘Don’t know’ answers

The last two charts (titled: Don't Know) represent the lack of technical knowledge of the respondents in both questionnaire. However, the charts show that the number of X's has been increased in the 2nd questionnaire in both *quality in-use* and *product quality* related characteristics.

To show how the results of the two questionnaires are distributed, we performed a statistical analysis using Fisher's Exact Test. For this purpose, we compared the five-scale percentages of the 1st and the 2nd questionnaires for each quality characteristic. The Null Hypothesis is: "*The participants in both* questionnaire*s have the same belief about the total quality of SharePoint at TEC*". If *p-value > 5%* then we cannot reject the null hypothesis. If *p-value < 5%* then we reject the null hypothesis.

Table 57 shows a sample of our data. The *p-value* is calculated for (A_1, A_2) columns, as well as for(B_1, B_2), and so on. Each two columns represent the data of a specific standard quality characteristic. In the most left column, the rows represent the rating scales.

Table 57: p-values obtained from Fisher's Test for three sample quality characteristics of both questionnaires

	Effectiveness		Efficiency		Satisfaction		Other values for other quality characteristics…
	1st	**2nd**	**1st**	**2nd**	**1st**	**2nd**	
	A1	A2	B1	B2	C1	C2	
Fully Supported	8	4	11	7	5	3	
Mostly Supported	18	7	20	10	19	6	
Fairly Supported	22	22	20	18	25	25	
Poorly Supported	20	19	16	20	13	31	
Not Supported	3	6	2	4	3	4	
Don't know	29	42	30	41	31	31	
Fisher's p-value	**9%**		**21%**		**1%**		

The p-values are shown in Table 58 for each quality characteristic.

Table 58: the Quality Characteristics and the p-values

Quality in-use	**Fisher p-value**
Effectiveness	9%
Efficiency	21%
Satisfaction	1%
Freedom from Risk	7%
Context Coverage	2%

Product Quality	**Fisher p-value**
Functional Suitability	5%
Performance efficiency	0%
Compatibility	0%
Usability	1%
Reliability	51%
Security	67%
Maintainability	0%
Portability	27%

Since the results show that in almost 50% of the quality characteristics the *p-value < 5%*, so we cannot accept the null hypothesis. We can conclude that the participants in the both questionnaires have different beliefs about the quality of SharePoint at TEC. We have already argued that the number of X's have increased in the 2nd questionnaire (non-technical users). Now the results of the fisher's test show that the two groups of the participants are somehow thinking differently about certain quality aspects of the software under study. The data in Table 59 show the questions and the number of Don't Know items (identified as X) in both questionnaires.

Table 59: Number of X's for each question in the both questionnaires – the reds are the outliers

Questions	**Number of X's (Questionnaires)**	
	1st	**2nd**
1- Undo availability: The consistent availability of an 'Undo' feature, even after writing to the database.	5	13
2- Audible warning: The capability of the product to present audible warning.	13	17
3- Help usability: The ability of the Help functions to provide adequate guidance on most issues.	4	11
4- Error-message coherence: Are the product's error messages clear and informative?	2	7
5- Clarity in user interfaces: Does the product provide clear, explicit and well-worded language within the user interfaces?	0	2
6- Mandatory fields: The product's ability to present a feature that distinguishes mandatory fields from non-mandatory ones.	1	4
7- Screen traceability: The product's ability to show where users are in the application on any/every screen.	1	6
8- Default values: The product's ability to allow users to define default values for desired fields using algorithms, equations, or business rules.	7	15
9- Screen color customization: The product's ability to allow users to adjust screen colors.	9	19
10- Minimalism: Does the product avoid redundant content or repeated feature entry-points?	2	10
11- Report customization: The product's ability to sort, filter, and customize standard reports.	7	11
12- Documentation accuracy: The product's ability to present the correct and complete user- and technical documentation.	8	19
13- Function key availability: The availability of function keys for frequently used menu items, or frequently used data entries.	7	18
14- Field word processing: The availability of advanced word processing functionality within alpha fields.	6	15
15- Learnability: How easily and quickly learned the product is for most users.	0	7
16- Controllability: The user's feeling of control over the proceedings of the software.	0	8
17- Secure remote accessibility: The ability of the product to be accessible remotely via a secure connection.	7	11
18- Mobile accessibility: The ability to access the product with mobile devices such as smartphones, tablets, etc.	14	18
19- Error handling agility: The product's ability to handle data-input errors quickly and easily.	6	9
20- Responsiveness: The product's ability to perform most functions at an acceptable speed.	0	3

Questions	Number of X's (Questionnaires)	
	1st	2nd
21- Satisfaction and interest level: The enjoyment or satisfaction of the users while using the product, and how engaged they feel with the product.	0	0
22- Expectancy: The product's ability to exceed expectations and meet needs that the user did not know he/she had.	2	3
23- Import/export options: The possibility to import/export data into the software using any type of file format.	8	17
24- Controlled modifiability: The product's capability to be "stable" and "reliable".	0	7
25- Backward compatibility: The product's ability to use files and data that were created with older versions of the product.	8	20
26- Interoperability: The ability of the product's components to interact with each other without undue delays or problems.	6	15
27- Authentication: The product's ability to identify users through log-in or other means.	2	6
28- Authorization: The product's ability to follow authorization protocols in allowing users to achieve their authorized level of access and use.	6	8
29- Privacy: The product's ability to protect data from being seen by unauthorized users.	9	13
30- Compliance: The product's ability to adhere to standards, conventions, or regulations in laws and similar prescriptions.	10	22
31-Access rights: The product's ability to assign different rights per user types in different sites.	9	13
32- Concurrency: The product's ability to perform multiple tasks in parallel without delays or problems.	5	11
33- Disaster recovery: The product's ability to recover from an unexpected failure without losing user information.	9	18

Although this table shows the number of X's, it is more illustrative and expressive if we use the percentages of X's instead of the number of X's. Therefore, for each question, we calculate the percentages as:

$$Percentage\ of\ X\ for\ Question\ Q_i = \frac{Number\ of\ X's\ received\ for\ Q_i}{\sum A, B, C, D, E, X} \qquad (4)$$

Where A to E and X are the code for the Likert scales presented in Table 55.

3.2. The Outliers

The numbers in Table 60 show the percentages of X over the total number of the received answers for each scale; from A to X. For example, for the question #1, 29% of the answers in the 1st questionnaire, were X. This value is changed 46% in the 2nd questionnaire.

Table 60: The percentage of X's received for each question in both questionnaires. The * denotes the outliers

Question	1st questionnaire (%)	2nd questionnaire (%)	Question	1st questionnaire (%)	2nd questionnaire (%)
1	29	46	18*	82	64
2*	76	61	19	35	32
3	24	41	20	0	11
4	12	25	21	0	0
5	0	7	22	12	11
6	6	14	23	47	61
7	6	21	24	0	26
8	41	54	25	47	71
9*	53	68	26	35	54
10	12	37	27	12	21
11	41	39	28	35	29
12	47	68	29	53	46
13	41	64	30*	59	79
14	35	54	31	53	46
15	0	26	32	29	39
16	0	30	33*	53	64
17	41	39			

Using this analysis method, we can identify the outliers systematically. For example, if we set the threshold to 40%, it means the questions that the number of X's are more than 40% of the total

number of received answers, are considered as an outlier. This rule is set for the both questionnaires. Thus, we will have 12 outliers for threshold=40%. We can control the threshold to get the optimum number of outliers. Obviously, the higher threshold values decrease the number of outliers. That is:

$$Threshold \propto \frac{1}{Number\ of\ outliers} \qquad (5)$$

Since the value of the threshold is used to clean up the both questionnaires from the outliers, selecting the optimum threshold value should be done precisely. If the threshold is too high, then lower number of outliers are identified, so many unrelated and vague questions that include lots of X will remain in the questionnaire. On the other hand, if the threshold is too low, then the higher number of outliers are identified, so many of valuable and meaningful questions may be removed from the questionnaire. In both cases, the inappropriate value of thresholds negatively affects the precision of the questionnaire. Table 61 shows the numeric relationship between the threshold and the number of outliers, as well as the outlier questions that should be removed based on the selected threshold.

Table 61: The identified outlier questions based on the given thresholds

Threshold	Number of outliers
40	12
45	10
50	**5**
55	3
60	2

It seems that the threshold=50 is good enough since it gives 5 outliers; neither too high nor too low. If we look into the questions that will be removed by threshold=50, we find out that those questions are somehow beyond the knowledge of end users in an organization. Thus, removing them from the questionnaires doesn't affect the precision of the questionnaire. Threshold=50 means that if we ignore the questions that at least 50% of their received answers are X, then we can consider 5 questions as an outlier and remove them from the list of questions. Table 62

provides the explanation for the reason why the participants do not know about the above questions identified as outliers. We categorize the reasons as:

- Not needed
- Not possible
- Don’t care
- Not happened so far
- Lack of knowledge

We removed the outlier questions from the list and then analyzed the results excluding the outliers. The numbers and the charts hereinafter are based on the refined data, i.e. all the data excluding the outliers.

Table 62 : Justifications for outliers

Outlier item	Reason	Explanation
2 - Audible warning	• Not possible	Since making the noise is not allowed in the office during the working hours, there is no speaker connected to the computers. So, the feature – if exists – would not be usable for them. That's why most of the users don't know about it.
9 - Screen color customization	• Not needed	Most of the users do not know if they can change the color of SharePoint screen items because each user works with a limited number of SharePoint items. So, it is not necessary to highlight or mark a specific item in the SharePoint pages using colors. This feature is not useful for the TEC users so they do not know about it.
18 - Mobile accessibility	• Don't care • Not needed	The users that need to connect to the SharePoint remotely, can do that because they can connect to their machines remotely and then they will have all the installed applications such as SharePoint, and so on. So, they do not need to know if SharePoint has a capability to be accessed remotely on the mobile devices. Other users do not need to connect remotely.
30 - Compliance	• Do not care	Only a few number of the employees are involved in defining, analyzing and revising the TEC's business processes and to make sure that they comply with global best practice, norms, and standards. Most the users are just the end users who only work on final forms and features. So, it is expected that assuring the compliance with the standards is not followed by most of the users and therefore they do not care about it.
33 - Disaster recovery	• Not happened so far	Many users have not any experience on any SharePoint disaster at TEC. Therefore, because it is not occurred until

	• Lack of knowledge	now, they have no knowledge about the reaction of SharePoint at this case.

Conclusion: Looking at the data from a higher level of abstraction, and the results of the Fisher's exact test reveal that different user groups have a different understanding about the quality of a specific software application. In our case, the users in development department who are known as technically knowledgeable users find the software as higher quality than the users in other departments of the company. The key point here is the "knowledge". Therefore, we can say: "the software quality from the users' point of view depends on the knowledge of the users about the software developments and quality, in general, and on the application under study, specifically". In our case, the skilled users in software technical departments claimed that the SharePoint has higher quality, than the other users working in other departments.

3.3. Where the knowledge lacks

The next step is to focus on the missing knowledge of the users in the 2nd questionnaire. For this purpose, we define three indicators that enable us to select the departments and the users who suffer from lack of knowledge. Then we retrieve the questions - i.e. quality measure elements - that are related to the users' missing knowledge.

3.3.1. The Indicators

In this stage, we attempt to answer to question: "*what is the missing Knowledge in the 2nd group of participants that causes them to select more X's than the 1st group?*". This question led us to analyze the Don't Know answers deeply among the 2nd questionnaire answers. In these analyses, the goal is to focus on the users that use SharePoint as one of their main daily tasks, while they have still ambiguities about SharePoint. Then we investigate on SharePoint quality elements that are missing for those users. For this purpose, we defined three indicators.

1. **SharePoint Involvement Percentage per TEC Department ($SPIP_D$):** SharePoint is used to manage the business processes at TEC. At the time of conducting our surveys, 17 TEC's business processes had been implemented in SharePoint. Given that we have a list of the

17 processes, and we know how the TEC departments are involved in certain processes, we define an indicator named "SharePoint Involvement Percentage" (SPIP). This indicator is aimed to illustrate how the TEC's departments are involved in SharePoint.

For department D, the SPIP is defined as:

$$SPIP_D = \frac{\sum processes\ that\ department\ D\ is\ involved\ in}{\sum processes\ available\ in\ SP} \times 100 \qquad (6)$$

For example, for the *Marketing* department which is involved in 10 SharePoint processes, the involvement is 59%:

$$SPIP_{Marketing} = \frac{\sum processes\ that\ Marketing\ is\ involved\ in}{\sum processes\ available\ in\ SP} \times 100 = \frac{10}{17} \times 100 = 59\%$$

That means in average, the Marketing department is involved in 59% of the TEC's processes. The data in Table 63 shows the SPIP values for all 12 departments.

Table 63: SharePoint Involvement Percentage (SPIP)

Department	$SPIP_D$		Department	$SPIP_D$
Testing	41%		Project Development	59%
Editing/Translation	53%		R&D	41%
Electronic data interchange	29%		Research Analyst	53%
Executive	41%		Selection Services	29%
Marketing	59%		Small business groups	29%
Pre-Sales	41%		Vendor Services	29%

2. **Applicability of the Questions (APL_Q):** not all the questions are applicable to all the departments. The APL_Q is a percentage that indicates the relationship between the questions and the departments. The data for associating the questions to TEC departments

are collected from various interviews: with department employees, with SharePoint administration team, department documents and reviewing the recorded tasks in SharePoint. Table 64 shows the associations between the questions and the departments; the letter 'Y' denotes an association between the given departments in the row with the given question in the column. Because of the lack of space in Table 64 only four sample questions denoted as Q_1 to Q_4 are presented.

Table 64: Association of four sample questions with the departments

Department	**Q_1**	**Q_2**	**Q_3**	**Q_4**
Testing	Y	N	Y	Y
Editing/Translation	Y	Y	Y	Y
EDI	Y	Y	Y	Y
Executive	Y	N	Y	Y
Marketing	Y	N	Y	Y
Pre-Sales	N	Y	Y	Y
Project Development	Y	Y	Y	Y
R&D	Y	Y	Y	Y
Research Analyst	Y	Y	Y	Y
Selection Services	Y	Y	Y	Y
Small business groups	N	Y	Y	Y
Vendor Services	N	Y	Y	Y
APL_Q (%)	**75**	**75**	**100**	**100**

In Table 65 presents the value of APL_Q for all the questions. The values show that there are some general questions that are related to all departments (such as Q_3, Q_4, Q_5, etc.), and the questions that are applicable for some of the departments. In the last row shows the values of APLQ. For example, the value of 75% for the question #1 indicates that the question #1 is applicable for 75% of the TEC's departments.

Table 65: Applicability of the questions for the departments

Question #	APL_Q	Question #	APL_Q	Question #	APL_Q
1	75	**10**	17	**19**	100
2	75	**11**	25	**20**	33
3	100	**12**	58	**21**	100
4	100	**13**	100	**22**	50
5	100	**14**	100	**23**	33
6	67	**15**	83	**24**	100
7	75	**16**	100	**25**	100
8	92	**17**	100	**26**	50
9	67	**18**	100	**27**	58
				28	33

Another indicator is defined to show how a specific department is involved into the questions. We call this indicator as INV_D, which is explained below.

3. **Involved departments (INV_D):** The values for this indicator can be extracted by looking at the department-question associations from the reverse point of view. The INV_D indicates a percentage of the questions that the department *D* is involved in. For example, the department '*Small Business Group (SBG)*' is involved in 50% of the questions:

$$INV_{SBG} = \frac{Number\ of\ questions\ that\ SBG\ department\ is\ involved\ in}{Total\ questions} = \frac{14}{28} \times 100 = 50\% \qquad (7)$$

Table 66 shows the INV values for all departments.

Table 66 : Involved Departments

Department	INV_D		Department	INV_D
Testing	93		Project Development	89
Editing/Translation	82		R&D	96
EDI	57		Research Analyst	75
Executive	82		Selection Services	61
Marketing	86		Small business groups	50
Pre-Sales	68		Vendor Services	57

3.4. Discussions

3.4.1. Question Applicability vs. Don't know

The goal here is to retrieve the questions that are applicable to many departments and at the same time, many users do not have knowledge about the quality element that lies in that question. It means we highlight the questions that both value of 'APL_Q' and the 'count of X' are high. However, we pay attention to two different types of X: 1) the answer X's which are applicable to the department, and 2) the answer X's that are not applicable. For this purpose, we created a matrix named Applicability-Data Matrix by combining the following two matrixes:

- The Applicability Matrix that shows if a specific question is related to a specific department. The rows of this matrix are the departments, the columns are the questions, and the elements are 'Yes' or 'No'. in this matrix, 'Yes' means that the given question is applicable for the given department, and 'No' means the given question is irrelevant for the given department.
- The Data Matrix that includes the answers of the users to the questionnaire. The rows of this matrix are the participants, the columns are the questions, and the element can be one of A, B, C, D, E, or X.

Thus, the rows of the Applicability-Data matrix are the departments, the columns are the questions and the elements will be pairs of (P_1, P_2), where:

$$P_1 = \{A, B, C, D, E, X\} \qquad P_2 = \{Yes, No\}$$

So, the format of the elements of the Applicability-Data matrix is defined as:

$$(P_1, P_2) = (response, applicability) \qquad (8)$$

This process was done for all the received user answers. Table 67 shows the Applicability-Data matrix for a sunset of departments, users, and questions.

Table 67: The Applicability-Data Matrix

Departments	**Users**	**Q_1**	**Q_2**	**Q_3**	**Q_4**	
Testing	U_{21}	D Yes	C No	B Yes	C Yes	
Testing	U_{27}	E Yes	C No	D Yes	C Yes	
Editing/Translation	U_{16}	X Yes	X Yes	B Yes	D Yes	
Editing/Translation	U_{20}	D Yes	D Yes	D Yes	C Yes	
Electronic Data Interchange	U_7	C Yes	D No	C Yes	C Yes	
Electronic Data Interchange	U_8	C Yes	D No	D Yes	C Yes	
Electronic Data Interchange	U_{25}	X Yes	D No	D Yes	D Yes	
Executive	U_9	E Yes	E No	D Yes	D Yes	
Marketing	U_6	E Yes	X Yes	C Yes	B Yes	
Marketing	U_{15}	E Yes	E Yes	D Yes	C Yes	
Pre-Sale	U_4	X No	X Yes	X Yes	C Yes	
The rest of the rows of the matrix…						*The rest of the columns of the matrix…*

In this matrix, the 'X Yes' element means the correspondent user (e.g. U_6 in Table 67) doesn't know about the point that lies in the correspondent question (e.g. Q_2 in Table 67), while the Q_2 is

applicable for the user U6. Another example; the ‘C Yes’ value shows U_{15} has responded C to the question Q_4 and the question Q_4 is applicable for the user U_{15}. Conversely, the ‘X No’ value shows that for example the U_4 doesn’t know about the question Q_1, but the point lies in Q_1 is not applicable for the user U_4. The results also show a strong negative correlation between the values of applicability (APL_Q) and the number of X of the questions (-0.80). This correlation can be interpreted as “the more the questions are applicable for the users, the less X we have received”. In the other words, the users that have fewer ambiguities about the quality elements that are applicable to them; their missing knowledge is mostly about the items that are not related to their daily tasks.

3.4.2. The knowledge missing items

Now we have enough data to analyze the questions that users do not know about them while they are relevant. We label the ‘knowledge missing’ questions as the ones that we received X for them while the question is applicable for the respondents. If the answer is X but the question is not applicable, we consider it as a “*do not* care” item. Table 68 shows all the questions and the correspondent ‘X Yes’ values. The higher values in this table are the ones that we call them ‘knowledge missing’ questions. The results demonstrate that the quality point in the knowledge missing items should be emphasized in the TEC Company. A training can be useful to increase the users’ knowledge, and this may help the users having a better perception of the SharePoint quality at TEC.

Table 68 : The questions with the number of 'X Yes' values categorized by 'Knowledge missing' and 'don't care' labels

Label: Knowledge missing	
Question Title	‘X Yes’
1 - Undo availability	8
2 - Help usability	12
3 - Error-message coherence	7
7 - Default values	11
8 - Minimalism	10
9 - Report customization	6

Label: Do not care	
Question Title	‘X Yes’
4 - Clarity in user interfaces	2
5 - Mandatory fields	4
6 - Screen traceability	1
10 - Documentation accuracy	2
12 - Field word processing	5
17 - Responsiveness	3

Table 68 : The questions with the number of 'X Yes' values categorized by 'Knowledge missing' and 'don't care' labels

Label: Knowledge missing	
11 - Function key availability	6
13 - Learnability	8
14 - Controllability	9
15 - Secure remote accessibility	10
16 - Error handling agility	9
21 - Controlled modifiability	8
24 - Authentication	6
25 - Authorization	8
26 - Privacy	8
27 - Access rights	7

Label: Do not care	
18 - Satisfaction and interest level	0
19 - Expectancy	3
20 - Import/export options	4
22 - Backward compatibility	5
23 - Inter-operability	2
28 - Concurrency	1

3.5. Correspondence with the reality

Although the participants of the 2nd questionnaire were less technical knowledgeable that the 1st questionnaire participants, we can identify the tech-savvy departments among all TEC departments. In this context, the employees of the tech-savvy departments are those who participated in the 2nd questionnaire and whose daily tasks are more technical and IT oriented than the other department employees. This identification was done by looking at their background educational field and certificates. Table 69 shows the list of tech-savvy departments among the other departments. It shows that the employees of the tech-savvy departments have selected less X's than the other employees. To distinguish the tech-savvy departments, we calculated the average number of 'X Yes' per participants, which shows the average of received X while the question is applicable for all participants. For example, in Testing department, in average, each participant selected 3.5 X's to the applicable questions.

Table 69 :Tech-savvy departments at TEC

Department	**Average number of 'X Yes' per participant**	**Employees Background**	**Label**
Testing	3.5	Engineering	Tech-Savvy
Editing/Translation	5.5	Humanities/Literature	
Electronic data interchange	2.3	Engineering + Business	Tech-Savvy
Executive	1.0	Engineering + Business	Tech-Savvy
Marketing	11.8	Business + Web Design	
Pre-Sales	10.0	Humanities/Business/Trade	
Project Development	6.5	Engineering + Business	
R&D	4.0	Engineering	Tech-Savvy
Research Analyst	5.5	Business	
Selection Services	6.5	Business	
Small business groups	6.0	Business	
Vendor Services	4.8	Business	

4. THREAT TO VALIDITY

There is a threat that the questions were answered by the employees of the departments that are not involved in SharePoint. For checking the validity and the consistency of our questionnaire, in this section, we show that the departments that are more involved in SharePoint, are involved in more questions. In other words, we show that the values of SPIPD and INVD are positively correlated. Table 70 shows the values for all the departments. It compares the amount that a given department is involved in TEC's processes, with the amount that the department is involved in the questions. For example, the Testing department is involved in 65% of TEC's processes in SharePoint while it is involved in 93% of the questions.

Table 70: SharePoint Involvement vs. Questions Involvement

Department	**SPIP (%)**	**INV (%)**
Testing	65	93
Editing/Translation	71	82
Electronic Data Interchange	47	57
Executive	65	82
Marketing	76	86
Pre-Sales	47	68
Project Development	82	89
R&D	88	96
Research Analyst	53	75
Selection Services	41	61
Small Business Groups	59	50
Vendor Services	35	57
Correlation	*0.82*	

The correlation value is *0.82* which is strong enough to support the claim that the departments which are strongly involved in TEC's processes in SharePoint have more related questions in the questionnaire.

5. CONCLUSION

In this study, the effects of participant's knowledge on the questionnaire results are addressed. The results of our systematic literature review show that the existence of this effect is controversial

among the researchers. To examine deeper, we conducted two surveys in a software consulting company. The surveys were targeted to evaluate the quality of Microsoft SharePoint from the end users' point of view at the company. To investigate the participants' knowledge, we focused on the number of 'Don't know' that we received from the participants. The results of our surveys approve that the participants' knowledge may affect the results of the surveys. According to the obtained results of the two surveys, we can conclude the following points. For simplification, we labeled the 'Don't know' as X in the text.

- *Questions should be applicable:* The questions that have more X, are those questions that are not applicable for most of the users. Thus, the more applicable the questions are, the less X are answered (according to the Table 67). It means that the participants are more knowledgeable. Participant's relevant knowledge plays a key role to decrease the number of X's.
- *The most beneficial participants should be cherry-picked:* For conducting a survey in software engineering, it is not enough to select appropriate questions or increase the number of participants regardless of their competencies. Our study shows that the participants should be cherry-picked based on their related knowledge. The more technical knowledgeable participants we select, the fewer X answers we receiv, and consequently, the result of the survey will be more valid.
- *Training is essential:* In a general perspective, the results reveal the idea that the questions that are applicable for most of the users while still many X's have been answered for them, can be considered as training subjects in the organization. The training needed items are those quality elements that are essential, but the users have not relevant knowledge about them, so the users need training on those subjects, and that knowledge may enhance the software quality in use.
- *Participants' knowledge of IT is essential:* The departments that have many X's are those that their employees are not tech-savvy. It reveals that increasing the general knowledge of IT of the employees or hiring the tech-savvy employees will affect the perception of the software product quality.

6. ACKNOWLEDGMENT

This work was supported by the Canadian Mitacs-Accelerate program [grant number IT08408]. We thank the TEC's employees who participated in our surveys and special thanks to Mehdi Aftahi, CTO of TEC for assistance and comments that greatly improved the manuscript.

7. REFERENCES

[1] R. Mirsalari and P. N. Robillard, "Industrial Validation of an Approach to Measure Software Quality," in *SEDE*, 2015, p. 6.

[2] L. Westfall, *The Certified Software Quality Engineer Handbook*. ASQ Quality Press; Har/Cdr edition, 2009.

[3] I. Standard, "ISO/IEC Systems and Software Engineering—Vocabulary (ISO/IEC/IEEE 24765:2010," 2015.

[4] Crosby Philip B., *Quality is free*. McGraw-Hill Science/Engineering/Math; 3 edition, 1979.

[5] J. M. Juran, *Juran's Quality Handbook*, Fifth Edit. New York, USA: McGraw-Hill, 1999.

[6] G. Schulmeyer and J. McManus, *Handbook of Software Quality Assurance*, 3rd Ed. Upper Saddle River, NJ: Prentice Hall PTR, 1998.

[7] L. Arksey, H., & O'Malley, "Scoping studies: towards a methodological framework," *Int. J. Soc. Res. Methodol.*, vol. 8, no. 1, pp. 19–32, 2005.

[8] H. L. Colquhoun, D. Levac, K. K. O'Brien, S. Straus, A. C. Tricco, L. Perrier, M. Kastner, and D. Moher, "Scoping reviews: Time for clarity in definition, methods, and reporting," *J. Clin. Epidemiol.*, vol. 67, no. 12, pp. 1291–1294, 2014.

[9] R. Mirsalari and P. N. Robillard, "Expected Software Quality Profile: A methodology and a case study," in *The 7th IEEE Annual Information Technology, Electronics & Mobile Communication Conference - IEMCON 2016*, 2016, p. 923.

[10] M. Kläs, C. Lampasona, and J. Münch, "Adapting software quality models: Practical challenges, approach, and first empirical results," in *Proceedings - 37th EUROMICRO Conference on Software Engineering and Advanced Applications, SEAA 2011*, 2011, pp. 341–348.

[11] J. Gottschick and H. Reste, "An empirical evaluation of the quality of interoperability

specifications for the web," in *Proceedings - 36th EUROMICRO Conference on Software Engineering and Advanced Applications, SEAA 2010*, 2010, pp. 398–405.

[12] I. Biscoglio and E. Marchetti, "An experiment of software quality evaluation in the audio-visual media preservation context," in *Proceedings - 2014 9th International Conference on the Quality of Information and Communications Technology, QUATIC 2014*, 2014, pp. 118–123.

[13] L. Aversano and M. Tortorella, "Analysing the Reliability of Open Source Software Projects," pp. 348–357, 2015.

[14] K. Lochmann, J. Ramadani, and S. Wagner, "Are comprehensive quality models necessary for evaluating software quality?," in *Proceedings of the 9th International Conference on Predictive Models in Software Engineering - PROMISE '13*, 2013, pp. 1–9.

[15] R. Hofman, "Behavioral economics in software quality engineering," *Empir. Softw. Eng.*, vol. 16, no. 2, pp. 278–293, 2011.

[16] L. Corral and I. Fronza, "Better Code for Better Apps: A Study on Source Code Quality and Market Success of Android Applications," in *Proceedings - 2nd ACM International Conference on Mobile Software Engineering and Systems, MOBILESoft 2015*, 2015, pp. 22–32.

[17] P. D. Anjali, "Empirical Validation of Website Quality Using Statistical and Machine Learning Methods," pp. 1–6, 2014.

[18] D. D. J. Suwawi, E. Darwiyanto, and M. Rochmani, "Evaluation of academic website using ISO/IEC 9126," *2015 3rd Int. Conf. Inf. Commun. Technol. ICoICT 2015*, pp. 222–227, 2015.

[19] R. Mirsalari and P. N. Robillard, "Industrial Validation of an Approach to Measure Software Quality," *Sede*, p. 6, 2015.

[20] L. Kumar and S. K. Rath, "Hybrid functional link artificial neural network approach for predicting maintainability of object-oriented software," *J. Syst. Softw.*, vol. 121, pp. 170–190, 2016.

[21] S. Chakrabarty and N. Chaki, "Quality Evaluation of Conceptual Level Object

Multidimensional Data Model," *Int. J. Comput. Appl.*, vol. 32, no. 3, p. 14, 2011.

[22] M. Pušnik, M. Heričko, Z. Budimac, and B. Šumak, "XML schema metrics for quality evaluation," *Comput. Sci. Inf. Syst.*, vol. 11, no. 4, pp. 1271–1290, 2014.

[23] Carnegie Mellon University., "Software Engineering Institute (SEI)," 1984. [Online]. Available: www.sei.cmu.edu.

[24] ISO/IEC, "ISO/IEC 25000 - Software Engineering — Software product Quality Requirements and Evaluation (SQuaRE)," 2005.

[25] E. H. Marinho and R. F. Resende, "Quality Factors in Development Best Practices," in *ICCSA*, 2012, pp. 632–645.

[26] ISO/IEC, "INTERNATIONAL STANDARD ISO/IEC 25010:2010," 2010.

[27] ISO/IEC, "INTERNATIONAL STANDARD ISO/IEC DIS 25021 — Quality Measure Element," 2011.

[28] T. Punter, M. Ciolkowski, B. Freimut, I. John, F. Iese, and D.- Kaiserslautern, "Conducting On-line Surveys in Software Engineering Characterizing surveys in SE," 2003.

[29] T. C. Lethbridge, S. E. Sim, and J. Singer, "Studying Software Engineers : Data Collection Techniques for Software Field Studies," *Empir. Softw. Eng.*, vol. 10, pp. 311–341, 2005.

[30] B. Kitchenham and S. L. Pfleeger, "Principles of Survey Research Part 5: Populations and Samples," *ACM SIGSOFT Softw. Eng. Notes*, vol. 27, no. 5, p. 17, 2002.

[31] B. a Kitchenham and S. L. Pfleeger, "Principles of Survey Research Part 3: Constructing a Survey Instrument," *ACM SIGSOFT Softw. Eng. Notes*, vol. 27, no. 2, p. 20, 2002.

[32] B. Kitchenham and S. L. Pfleeger, "Principles of survey research part 4: questionnaire evaluation," *ACM SIGSOFT Softw. Eng. Notes*, vol. 27, no. 3, p. 20, 2002.

[33] B. Kitchenham and S. L. Pfleeger, "Principles of survey research part 6: Data Analysis," *ACM SIGSOFT Softw. Eng. Notes*, vol. 28, no. 2, p. 24, 2003.

[34] B. Kitchenham and S. L. Pfleeger, "Principles of Survey Research," *ACM SIGSOFT Softw. Eng. Notes*, vol. 26, no. 6, p. 16, 2001.

[35] B. a Kitchenham and S. L. Pfleeger, "Principles of Survey Research Part 2 : Designing a

Survey Sample size Experimental designs," *Softw. Eng. Notes*, vol. 27, no. 1, pp. 18–20, 2002.

[36] H. Al-Kilidar, K. Cox, and B. Kitchenham, "The use and usefulness of the ISO/IEC 9126 quality standard," in *International Symposium on Empirical Software Engineering.*, 2005, pp. 122–128.

[37] R. E. Al-qutaish, "A Maturity Model of Software Product Quality," *Res. Pract. Inf. Technol.*, vol. 43, no. 4, pp. 307–328, 2010.

[38] E. Van Veenendaal, R. Hendriks, and R. Van Vonderen, "Measuring Software Product Quality," *SQP*, vol. 5, no. 1, pp. 6–13, 2002.

[39] C. Wohlin, P. Runeson, M. Host, Magnus C. Ohlsson, Bjorn Regnell, and Anders Wesslen, *Experimentation in Software Engineering*. 2012.

[40] P. Torino and P. Torino, "A State-of-the-Practice Survey of Risk Management in Development with Off-the-Shelf A State-of-the-Practice Survey on Risk Management in Development with Off- The-Shelf Software Components," vol. 34, no. April, pp. 271–286, 2016.

[41] A. Nugroho and M. R. V. Chaudron, "A survey into the rigor of UML use and its perceived impact on quality and productivity," *Proc. Second ACM-IEEE Int. Symp. Empir. Softw. Eng. Meas. - ESEM '08*, p. 90, 2008.

[42] J. Soini, "A Survey of Metrics Use in Finnish Software Companies," *2011 Int. Symp. Empir. Softw. Eng. Meas.*, pp. 49–57, 2011.

[43] W. Chen, J. Li, J. Ma, R. Conradi, J. Ji, and C. Liu, "A survey of software development with open source components in Chinese software industry," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 4470 LNCS, pp. 208–220, 2007.

[44] J. Li, J. Ma, R. Conradi, W. Chen, J. Ji, and C. Liu, "A survey on the business relationship between Chinese outsourcing software suppliers and their outsourcers," *Proc. - Asia-Pacific Softw. Eng. Conf. APSEC*, pp. 470–477, 2007.

[45] E. Petrinja, A. Sillitti, and G. Succi, "Adoption of OSS Development Practices by the

Software Industry: A Survey," *Open Source Syst. Grounding Res.*, vol. 365, pp. 233–243, 2011.

[46] A. Macphail, T. Hainey, and T. M. Connolly, "Applying Mlearning in Software Engineering Education : a Survey of Mobile Usage," *Mob. Learn.*, no. iii, 2012.

[47] M. Usman, E. Mendes, and J. Börstler, "Effort estimation in Agile software development: A survey on the state of the practice," *ACM Int. Conf. Proceeding Ser.*, vol. 27–29–Apri, 2015.

[48] R. N. Memon, S. S. Salim, and R. Ahmad, "Identifying research gaps in requirements engineering education: An analysis of a conceptual model and survey results," *2012 IEEE Conf. Open Syst.*, pp. 1–6, 2012.

[49] Y. Cerqueira, S. R. De Lemos, Y. C. . Cavalcanti, P. A. . Da Mota Silveira Neto, I. . Do Carmo Machado, E. S. . De Almeida, and S. R. . De Lemos Meira, "Towards Understanding Software Change Request Assignment : a survey with practitioners," *ACM Int. Conf. Proceeding Ser.*, pp. 195–206, 2013.

[50] C. Palomares, C. Quer, and X. Franch, "Requirements reuse and requirement patterns : a state of the practice survey," *Empir. Softw. Eng.*, 2016.

[51] David Ameller, M. Galster, P. Avgeriou, and X. Franch, "A survey on quality attributes in service-based systems," *Softw. Qual. J.*, vol. 24, no. 2, pp. 337–364, 2016.

[52] H. Tsuji, A. Sakurai, K. Yoshida, A. Tiwana, and A. Bush, "Questionnaire-Based Risk Assessment Scheme for," pp. 114–127.

[53] H.-C. Huang, "Freemium business model: construct development and measurement validation," *Internet Res.*, vol. 26, no. 3, pp. 604–625, 2016.

[54] D. Tofan, M. Galster, P. Avgeriou, and D. Weyns, "Software engineering researchers' attitudes on case studies and experiments: An exploratory survey," *Eval. Assess. Softw. Eng. (EASE 2011), 15th Annu. Conf.*, no. 638, pp. 91–95, 2011.

[55] O. Albayrak, "Instructor's Acceptance of Games Utilization in Undergraduate Software Engineering Education: A Pilot Study in Turkey," *2015 IEEE/ACM 4th Int. Work. Games Softw. Eng.*, pp. 43–49, 2015.

[56] D. Budgen, B. A. Kitchenham, S. M. Charters, M. Turner, P. Brereton, and S. G. Linkman, “Presenting software engineering results using structured abstracts: A randomised experiment,” *Empir. Softw. Eng.*, vol. 13, no. 4, pp. 435–468, 2008.

[57] F. Q. B. Da Silva and A. C. C. Frana, “Towards understanding the underlying structure of motivational factors for software engineers to guide the definition of motivational programs,” *J. Syst. Softw.*, vol. 85, no. 2, pp. 216–226, 2012.

[58] I. Erfurth and W. R. Rossak, “A look at typical difficulties in practical software development from the developer perspective A field study and a first solution proposal with UPEX,” *Proc. Int. Symp. Work. Eng. Comput. Based Syst.*, pp. 241–248, 2007.

[59] N. Fenton, M. Neil, W. Marsh, P. Hearty, Ł. Radliński, and P. Krause, “On the effectiveness of early life cycle defect prediction with Bayesian nets,” *Empir. Softw. Eng.*, vol. 13, no. 5, pp. 499–537, 2008.

[60] I. Garcia, C. Pacheco, and P. Sumano, “Use of questionnaire-based appraisal to improve the software acquisition process in small and medium enterprises,” *Stud. Comput. Intell.*, vol. 150, pp. 15–27, 2008.

[61] J. Hutchinson, J. Whittle, M. Rouncefield, and S. Kristoffersen, “Empirical assessment of MDE in industry,” *2011 33rd Int. Conf. Softw. Eng.*, pp. 471–480, 2011.

[62] M. V. Kosti, R. Feldt, and L. Angelis, “Personality, emotional intelligence and work preferences in software engineering: An empirical study,” *Inf. Softw. Technol.*, vol. 56, no. 8, pp. 973–990, 2014.

[63] T. Sedano, “Code readability testing, an empirical study,” *Proc. - 2016 IEEE 29th Conf. Softw. Eng. Educ. Training, CSEEandT 2016*, pp. 111–117, 2016.

[64] P. . Diebold, A. . Vetró, and D. . Méndez Fernández, “An Exploratory Study on Technology Transfer in Software Engineering,” *Int. Symp. Empir. Softw. Eng. Meas.*, vol. 2015–Novem, pp. 86–95, 2015.

[65] A. Forward and T. C. Lethbridge, “Problems and Opportunities for Model-Centric Versus Code-Centric Software Development: A Survey of Software Professionals,” *Proc. 2008 Int. Work. onModels Softw. Eng.*, pp. 27–32, 2008.

[66] J. Ji, J. Li, and R. Conradi, “Some lessons learned in conducting software engineering surveys in China,” *ESEM’08 Proc. 2008 ACM-IEEE Int. Symp. Empir. Softw. Eng. Meas.*, pp. 168–177, 2008.

[67] J. M. Rojas, G. Fraser, and A. Arcuri, “Automated Unit Test Generation During Software Development: A Controlled Experiment and Think-aloud Observations,” pp. 338–349, 2015.

[68] P. Karpati, Y. Redda, A. L. Opdahl, and G. Sindre, “Comparing attack trees and misuse cases in an industrial setting,” *Inf. Softw. Technol.*, vol. 56, no. 3, pp. 294–308, 2014.

[69] J. P. Campos, J. L. Braga, A. M. de Resende, and C. H. Os’orio Silva, “Identification of Aspect Candidates by Inspecting Use Cases Descriptions,” *SIGSOFT Softw. Eng. Notes*, vol. 35, no. 4, pp. 1–9, 2010.

[70] L. Prechelt and M. Liesenberg, “Design patterns in software maintenance: An experiment replication at Freie Universit??t Berlin,” *Proc. - 2011 2nd Int. Work. Replication Empir. Softw. Eng. Res. RESER 2011*, pp. 1–6, 2012.

[71] M. Haddara and A. Elragal, “ERP adoption cost factors identification and classification: a study in SMEs,” *Int. J. Inf. Syst. Proj. Manag.*, vol. 1, no. 2, pp. 5–21, 2013.

[72] A. Jedlitschka, “Evaluating a model of software managers’ information needs,” *Proc. 2010 ACM-IEEE Int. Symp. Empir. Softw. Eng. Meas. - ESEM ’10*, p. 1, 2010.

[73] M. Sensalire, P. Ogao, and A. Telea, “Evaluation of software visualization tools: Lessons learned,” *2009 5th IEEE Int. Work. Vis. Softw. Underst. Anal.*, pp. 19–26, 2009.

[74] M. Schmidberger and B. Brugge, “Need of Software Engineering Methods for High Performance Computing Applications,” *Parallel Distrib. Comput. (ISPDC), 2012 11th Int. Symp.*, pp. 40–46, 2012.

[75] D. Galin, *Software Quality Assurance From theory to implementation*. 2004.

[76] S. H. Kan, *Metrics and Models in Software Quality Engineering*. Addison-Wesley Longman Publishing Co., Inc., 2002.

[77] Rohaiza Abd. Rokis, “Youth Employability and Work Attitudes,” *Int. J. Sci. Commer. Humanit.*, vol. 2, no. 5, p. 13, 2014.

[78] G. Vigderhous, *The Level of Measurement and Permissible Statistical Analysis in Social Research*, vol. 20, no. 1. 1977.

[79] U. Jakobsson, "Statistical presentation and analysis of ordinal data in nursing research," *Scand. J. Caring Sci.*, vol. 18, no. 4, pp. 437–440, 2004.

[80] IEEE Computer Society, "IEEE Standard for a Software Quality Metrics Methodology - IEEE Std 1061™-1998 (R2009)," vol. 1998, 2009.

[81] H. Jung and S. Kim, "Measuring Software Product Quality: A Survey of ISO/IEC 9126," *IEEE Softw.*, pp. 88–92, 2004.

[82] L. M. Lozano, E. García-Cueto, and J. Muñiz, "Effect of the Number of Response Categories on the Reliability and Validity of Rating Scales," *Methodol. Eur. J. Res. Methods Behav. Soc. Sci.*, vol. 4, no. 2, pp. 73–79, Jan. 2008.

[83] C. Alzola and F. Harrell, "An Introduction to S and the Hmisc and Design Libraries," 2006. [Online]. Available: http://her.gr.distfiles.macports.org/mirrors/CRAN/doc/contrib/Alzola+Harrell-Hmisc-Design-Intro.pdf. [Accessed: 29-Jan-2015].

[84] "Technology Evaluation Centers." [Online]. Available: http://www.technologyevaluation.com/. [Accessed: 18-Apr-2017].

[85] J. Baroudi, M. Olson, and B. Ives, "An empirical study of the impact of user involvement on system usage and information satisfaction," *Commun. ACM*, vol. 29, no. 3, pp. 232–238, 1986.

[86] Software Engineering Institute, "CMMI® for Development, Version 1.3," 2010.

[87] IEEE Computer Society, *SWEBOK Guide*. 2014.

[88] S. Hansen and J. Rennecker, "Getting on the same page: Collective hermeneutics in a systems development team," *Inf. Organ.*, vol. 20, no. 1, pp. 44–63, Jan. 2010.

[89] O. Braud, "Facteurs decisionnels pour l'iplantation d'un ERP dans les PME : Le role de l'evaluation des benefices tangibles et intangibles," 2008.

[90] A. J. Albrecht, "Measuring application development productivity," *IBO Conf. Appl. Dev.*, vol. 10, pp. 83–92, 1979.

[91] R. Alvaro, “Framework for a global quality evaluation of a website,” *Online Inf. Rev.*, vol. 36, no. 3, pp. 347–382, 2012.

[92] H. Yang, “Measuring software product quality with ISO standards base on fuzzy logic technique,” *Adv. Intell. Soft Comput.*, vol. 137 AISC, pp. 59–67, 2012.

[93] P. Tomas, M. J. Escalona, and M. Mejias, “Open source tools for measuring the Internal Quality of Java software products. A survey,” *Comput. Stand. Interfaces*, vol. 36, no. 1, pp. 244–255, 2013.

[94] M. Kwiatkowski and C. Verhoef, “Recovering management information from source code,” *Sci. Comput. Program.*, vol. 78, no. 9, pp. 1368–1406, 2013.

[95] S. Lehtonen, “Metrics for Gerrit code reviews,” no. May, pp. 31–45, 2015.

[96] B. H. Layne, J. R. Decristoforo, and D. Mcginty, “Electronic versus traditional student ratings of instruction,” *Res. High. Educ.*, vol. 40, no. 2, pp. 221–232, 1999.

[97] B. Kitchenham, O. Pearl Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, “Systematic literature reviews in software engineering - A systematic literature review,” *Inf. Softw. Technol.*, vol. 51, no. 1, pp. 7–15, 2009.

[98] Ameller David, Galster Matthias, Paris Avgeriou, and Franch Xavier, “A survey on quality attributes in service-based systems,” *Softw. Qual J*, 2016.

[99] K. Dullemond and B. van Gameren, “What Distributed Software Teams Need to Know and When: An Empirical Study,” *Glob. Softw. Eng. (ICGSE), 2013 IEEE 8th Int. Conf.*, pp. 61–70, 2013.

[100] J. M. Rojas, G. Fraser, and A. Arcuri, “Automated unit test generation during software development: a controlled experiment and think-aloud observations,” *Proc. 2015 Int. Symp. Softw. Test. Anal. - ISSTA 2015*, pp. 338–349, 2015.

[101] A. Jedlitschka, “Evaluating a model of software managers’ information needs,” *Proc. 2010 ACM-IEEE Int. Symp. Empir. Softw. Eng. Meas. - ESEM ’10*, p. 1, 2010.

[102] A. Nugroho and C. F. J. Lange, “On the relation between class-count and modeling effort,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 5002 LNCS, pp. 93–104, 2008.

APPENDIX D – QUESTIONNAIRE: SOFTWARE QUALITY PROFILE – END USERS

How long have you been working in your current job position?

Your name and email address:

1- The consistent availability of an 'Undo' feature, even after writing to the database.

2- The product's ability to provide audible warnings.

3- The ability of the Help functions to provide adequate guidance on most features and issues.

4- The product's error messages should be clear and informative.

5- Within the user interface, the product should provide clear, explicit and well-worded language.

6- The product's ability to present a feature that distinguishes mandatory fields from non-mandatory ones.

7- The product's ability to show where users are in the application on any/every screen.

8- The product's ability to allow users to define default values for desired fields using algorithms, equations, or business rules.

9- The product's ability to allow users to adjust screen colors.

10- The product should avoid redundant content or repeated feature entry-points.

Example: ideally users should not be offered specific functionality under two or more different labels in two or more places in the software.

11- The product's ability to sort, filter, and customize standard reports.

Custom filtering allows software users to provide query criteria and apply operators in a preferred order of execution. The software may support Boolean "AND/OR/NOT" as custom conditions. The software may also be able to group and summarize the report data by

various conditions. Example: First sort by (STUDENT_ID) and (LAST_NAME) ascending, then filter on column (YEAR > 2010 and STATUS==“pass”) then Group by “DEPARTMENT_ID”.

12- The product’s ability to present the correct and complete user- and technical documentation.

13- The availability of Function keys (F1, F2, …, F12) for frequently used menu items, or frequently used data entries.

14- The availability of advanced word processing functionality within alpha fields.

15- The product’s ability to be learned easily and quickly, for most users.

The software is learnable when it uses the shape signals, help the user at the moment, no need for navigation in help resourcing, it avoids distraction, wizards carrying out the operations, it uses an appropriate language, it provides feedback messages that a user command has succeeded or it advises of failure, it makes commands and menu options highly visible and easy to find, etc.

16- The user's feeling of control over the proceedings of the software.

For example, if the software product uses a "wizard" or "progress bar" to show the progress of a process, this contributes to a feeling of control for the user.

17- The ability of the product to be accessible remotely via a secure connection.

Managing the tasks on your own computer from afar is possible when the software product is accessible remotely and when you have a secure remote desktop utility to rely on. Example: With Remote Desktop Connection, you can connect to a computer running Windows from another computer running Windows that's connected to the same network or to the Internet. You can then use all of your work computer's programs, files, and network resources from your home computer, as if you’re sitting in front of your computer at work.

18- The ability to access the product with mobile devices such as smart phones, tablets, etc.

19- The product’s ability to handle data-input errors quickly and easily.

The software may check for range/format/logic errors entered by the users. For example, does the product flag the user if she or he has entered text in a field that requires a number? Similarly, does it flag the user if he or she has entered data that falls outside an allowable range? Or does it flag the user if she or he hasn't respected a specific formatting rule, such as the format "A9A 9A9" for the postal codes in Canada?

20- The product's ability to perform most functions at an acceptable speed.

21- The enjoyment or satisfaction of the users while using the product, and how engaged they feel with the product.

22- The product's ability to exceed expectations and meet needs that the user did not know he/she had.

23- The possibility to import/export data into the software using any type of file formats.

Example: Importing data from an Excel file, Word file, XML file, CSV file, etc.

24- The product's capability to be "stable" and "reliable".

Example: we have an app, it works perfectly, aside from it is crashing every 5 minutes, but it backs to track instantly without data loss. We say the app is reliable, but not stable. I can rely on it since it does not miss data, and it is working correctly, despite it is not stable since it crashes periodically. In fact, the "internet" is basically that. It is far from stable, connections drop and reappear, packets are lost, and all kinds of other unstable things happen. However, it is reliable! It never happens if you ask to see Yahoo.com, but the internet brings you Google.com!

25- The product's ability to use files and data that were created with older versions of the product.

Example: A backward-compatible word processor, for instance, allows you to edit documents created with a previous version of the program.

26- The ability of the product's components to interact with each other without undue delays or problems.

Example: In a social network, when a user (Alice) searches for another user (Bob) or when adding Bob to her contact list, Bob will appear offline to Alice until Bob accepts Alice's request to add him.

27- The product’s ability to identify users through log-in or other means.

28- The product’s ability to follow authorization protocols in allowing users to achieve their authorized level of access and use.

Example: The students are not authorized to view certain web pages dedicated to professors and administration.

29- The product's ability to protect data from being seen by unauthorized users.

30- The product’s ability to adhere to standards, conventions, or regulations in laws and similar prescriptions.

Example: Banking software will be compliant with W3 guidelines for language protocols. Tax preparation software packages should be compliant with federal and provincial tax regulations.

31- The product’s ability to assign different rights per user types in different sites.

Access rights help to protect the IT system and the data stored on the system by restricting who can do what. Example: Your school. There are students, teachers, and the network staff. As a student, you can log on, access your own files and change them. The teachers can do more. They might be able to access all of the students' user areas and open, copy or move files. They can put files into the shared folder for students to use. Most teachers, however, often can't access other teachers' areas. They cannot change the system settings. They cannot add or delete users on the system. Network staff are responsible for the upkeep of the network and so have may have 'admin' rights to the system. This means they can do just about everything on the system. They can install new software, change system settings, and add/delete users. They can access everyone's files and folders.

32- The product’s ability to perform multiple tasks in parallel without delays or problems.

Example: if a calculation takes 30 minutes to be finished, does the software allow the user to work on another job while the calculation is in the process.

33- The product's ability to recover from an unexpected failure without losing user information.

Example: in the case of a power outage, database crash, or hard disk crash, the software is capable of preventing loss of data or recovering the lost data.

- Please give us your feedback

Your opinion is very important for us. We appreciate your feedback and will use it to evaluate changes and make improvements in the questionnaire.

Vos commentaires en français sont également les bienvenus.

APPENDIX E – QUESTIONNAIRE: SOFTWARE QUALITY PROFILE – POWER USERS

How long have you been working in your current job position?

Your name and email address:

1- The capability to install and configure the product in various ways or places.

Custom installation is a type of installation that allows the user to specify certain installation settings and options, such as which components will be installed.

2- The capacity of the product to remove files and release resources when uninstalling.

Uninstallation: To remove an application from a computer. Uninstalling removes all files that were added to the computer when the application was initially installed. In addition, it might also remove files that were subsequently generated by the application.

3- The ability of the product to be accessible remotely via a secure connection.

Managing the tasks on your own computer from afar is possible when the software product is accessible remotely and when you have a secure remote desktop utility to rely on. Example: With Remote Desktop Connection, you can connect to a computer running Windows from another computer running Windows that's connected to the same network or to the Internet. You can then use all of your work computer's programs, files, and network resources from your home computer, as if you are sitting in front of your computer at work.

4- The ability to access the product with mobile devices such as smart phones, tablets, etc.

5- The ability to expand and customize “fields” without accessing the source code.

Example: The user is able to add fields to maintenance records (e.g. customer, vendor, inventory and employee) and selected transaction records (e.g. sales order table, purchase order table, work order table, etc.).

6- The ability to customize “business rules” implemented in the software without accessing the source code.

Example: The user is able to change "purchasing" function in an organization, or change the calculation of annual tax without accessing the source code.

7- The ability to add new “menu items”, or modify the existing ones without accessing the source code.

8- The product’s ability to perform most functions at an acceptable speed.

9- The product’s ability to handle heavier demands for processing over an extended period without losing any of the performance measures.

Good endurance is evident if the system does not crash in spite of constantly and steadily increasing load.

10- The ability of the product’s Help functions to provide adequate guidance on most issues.

11- The product’s ability to provide audible warnings.

12- The product’s ability to sort, filter, and customize standard reports.

Custom filtering allows software users to provide query criteria and apply operators in a preferred order of execution. The software may support Boolean “AND/OR/NOT” as custom conditions. The software may also be able to group and summarize the report data by various conditions. Example: First sort by (STUDENT_ID) and (LAST_NAME) ascending, then filter on column (YEAR > 2010 and STATUS==“pass”) then Group by “DEPARTMENT_ID”.

13- The product’s ability to present the correct and complete user- and technical documentation.

14- The possibility to import/export data into the software using any type of file formats.

Example: Importing data from an Excel file, Word file, XML file, CSV file, etc.

15- The availability of development tools for the technical power user to add features in the future.

Does the product provide the tools that enable technical power users to add new features or modify the existing ones to meet the new requirements, without referring the job to the original software providers or maintainers?

16- The product's capability to be "stable" and "reliable".

Example: we have an app, it works perfectly, aside from it is crashing every 5 minutes, but it backs to track instantly without data loss. We say the app is reliable, but not stable. I can rely on it since it does not miss data, and it is working correctly, despite it is not stable since it crashes periodically. In fact, the "internet" is basically that. It is far from stable, connections drop and reappear, packets are lost, and all kinds of other unstable things happen. However, it is reliable! It never happens if you ask to see Yahoo.ca, but the internet brings Google.ca!

17- The product’s ability to use files and data that were created with older versions of the product.

Example: A backward-compatible word processor, for instance, allows you to edit documents created with a previous version of the program.

18- The ability of the product's components to interact with each other without undue delays or problems.

Example: In a social network, when a user (Alice) searches for another user (Bob) or when adding Bob to her contact list, Bob will appear offline to Alice until Bob accepts Alice's request to add him.

19- The ability to observe the internal states of the software when needed, using a bug monitoring feature.

Example: Through internal monitoring, a user can determine the cause of errors, faults or failures, or the progress of internal operations of the software.

20- The product’s ability to scale up, out, or down.

Example: Software is considered scalable if it can be moved from a smaller to a larger operating system without affecting factors like user response time.

21- The product’s ability to remain stable when computational limits are being challenged, such as when low disk space is encountered.

The goal is to ensure the product does not crash in conditions of insufficient computational resources. Example: Microsoft Word has a consecutive character limit 65,535. Once it reaches that limit, it refuses to accept more data. It therefore has good stress handling, because it reaches its limit without destabilizing.

22- The product’s ability to recover from an unexpected failure without losing users’ information.

Example: in the case of a power outage, database crash, or hard disk crash, the software is capable of preventing the loss of data or recovering lost data.

23- The product’s ability to handle upgrades without problems.

A software upgrade generally refers to any major upgrade to the software that adds significant changes to the program.

24- The product’s ability to identify users through log-in or other means.

Authentication is used whenever you want to know exactly who is using your software. Example: Commercial websites such as Amazon.ca require people to log in before buying products, so they know exactly who their purchasers are.

25- The product’s ability to follow authorization protocols in allowing users to achieve their authorized level of access and use.

Example: The students are not authorized to view certain web pages dedicated to professors and administration.

26- The product's ability to protect data from being seen by unauthorized users.

Encryption should be used whenever people are giving out personal information to for example register for something or buy a product. Doing so ensures the person’s privacy during the communication.

27- The product’s ability to adhere to standards, conventions, or regulations in laws and similar prescriptions.

Example: Banking software will be compliant with W3 guidelines for language protocols. Tax preparation software packages should be compliant with federal and provincial tax regulations.

28- The product’s ability to assign different rights per user types in different sites.

Access rights help to protect the IT system and the data stored on the system by restricting who can do what. Example: Your school. There are students, teachers, and the IT/network staff. As a student, you can log on, access your own files and change them. The teachers can do more. They might be able to access all of the students' user areas and open, copy or move files. They can put files into the shared folder for students to use. Most teachers, however, often can't access other teachers' areas. They cannot change the system settings. They cannot add or delete users on the system. Network staff are responsible for the upkeep of the network and so have may have 'admin' rights to the system. This means they can do just about everything on the system. They can install new software, change system settings, and add/delete users. They can access everyone's files and folders.

29- The product’s ability to force users to change passwords on a periodic basis.

Most government, large corporations, and even educational institutions want their users to change passwords after a certain time period. This provides an additional security layer to protect their network and servers.

30- The product’s ability to set constraints on the structure of the passwords.

Example: not less than 8 letters, must use lower and upper-case alphabets, must use one number, etc.

31- The product’s ability to record and manage log-in attempts.

Recording the log-in attempts is useful to monitor the excessive failed login attempts from a single IP address or range of addresses closely. This record may detect the hackers' attacks.

32- The product’s ability to provide password protection for individual fields, features, or menu items.

Example: the software might provide a facility to set a password on specific components such as fields, features, or menu items, although the user has entered the main password at the first login page.

33- The product's ability to encrypt the password during transmission.

34- The product's ability to support multilevel authentication.

In multi-level authentication, a user makes use of a primary username and password to log into his or her account, and then may be asked to enter a secondary password to perform some other actions, such as authorizing a transaction.

35- The product's adherence to the principle of minimalism, by not containing any redundant content or feature entry-points.

Example: ideally users should not be offered specific functionality under two or more different labels in two or more places in the software.

36- The degree of clarity the product offers in its user interfaces as expressed by explicit, unambiguous, and correct language and directions.

37- The degree to which the product's error messages are clear and informative.

- Please give us your feedback

Your opinion is very important for us. We appreciate your feedback and will use it to evaluate changes and make improvements in the questionnaire.

Vos commentaires en français sont également les bienvenus.